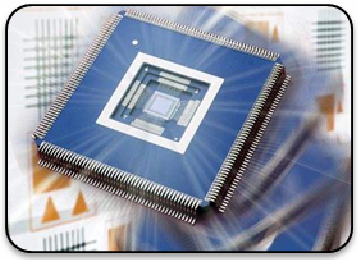


DECODIFICADORES
DEMULTIPLEXORES

Grupo 6:
Lara Tuero Terán
Elisa Pajarón López
Eduardo Ruiz Terrazas

Decodificadores

- Son circuitos combinacionales cuya función es inversa a la del codificador, es decir, convierte un código binario de entrada (natural, BCD, etc.) de entrada A de N bits en M líneas de salida Oi (donde N puede ser cualquier entero y M es un entero menor o igual a 2^N).

- **Para cada dato binario de entrada Ai se fija una única salida Oi a 1, cuyo índice i corresponde al valor binario del dato de entrada.**

Decodificadores

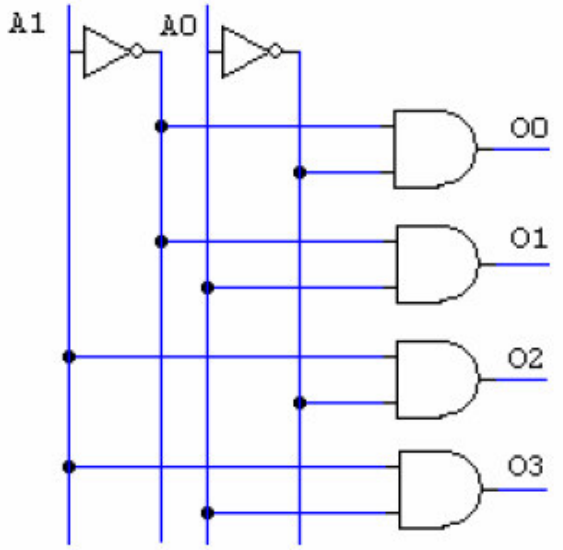
- Tenemos un decodificador de 2 entradas con 2^2=4 salidas (DEC 2 a 4). Su funcionamiento sería el que se indica en la siguiente tabla, donde se ha considerado que las salidas se activen con un "uno" lógico:

2 a 4 DEC

A1	A0	O0	O1	O2	O3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

$O0 = \overline{A1}\ \overline{A0}$ $O1 = \overline{A1}\ A0$

$O2 = A1\ \overline{A0}$ $O3 = A1\ A0$

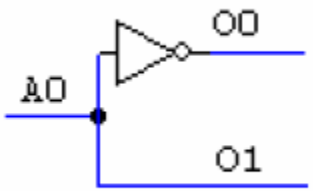


- En el caso de que a la entrada tengamos un “2” (1 0), tendríamos habilitada la salida O2.

Decodificadores

- Un decodificador 1 a 2 (DEC 1 a2) puede realizarse con un solo inversor:

A0	O0	O1
0	1	0
1	0	1



  - Para una entrada 0, tenemos la salida O0
  - Para una entrada 1, tenemos la salida O1

Decodificadores

- Todas las entradas de las puertas AND están a nivel alto, luego la salida que tendremos de ella estará a nivel ALTO también. Este tipo de decodificadores se les conoce como decodificadores **ACTIVADO A ALTO**, pero también es posible implementar un decodificador con salida en nivel BAJO. En lugar de utilizar puertas AND pueden utilizarse para ello puertas NAND e inversores a la salida si queremos obtener el mismo resultado. En su caso se le conoce como decodificador **ACTIVADO A BAJO** y se considera que las salidas se activan con un “cero” lógico. Este último tipo de decodificador suele ser **el más**

Decodificadores

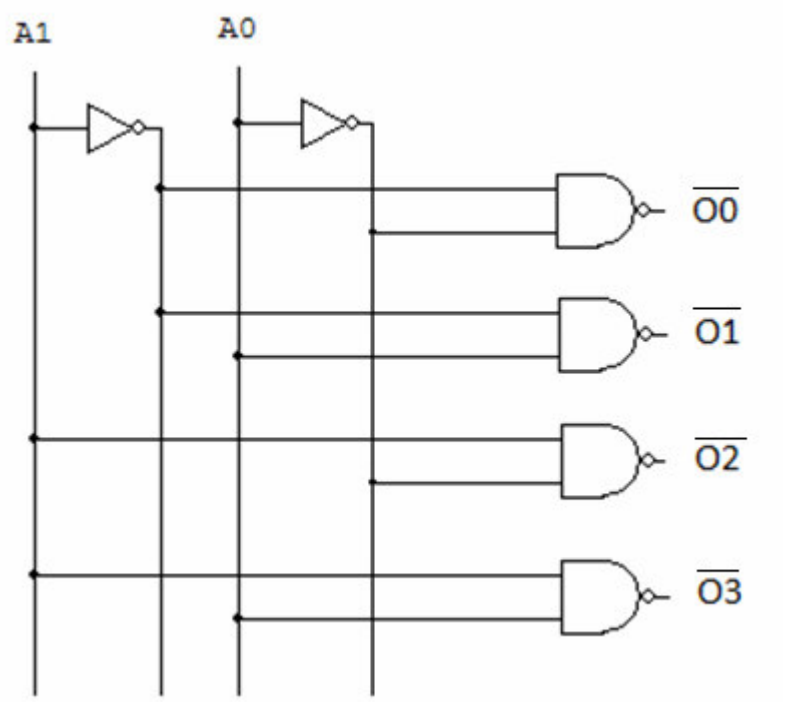
- DEC 2 a 4. Activado a bajo (Sin inversores a la salida)

2 a 4 DEC

A1	A0	$\overline{O0}$	$\overline{O1}$	$\overline{O2}$	$\overline{O3}$
0	0	0	1	1	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	1	1	1	0

$\overline{O0} = \overline{A1}\ \overline{A0}$ $\quad\quad$ $\overline{O1} = \overline{A1}\ A0$

$\overline{O2} = A1\ \overline{A0}$ $\quad\quad$ $\overline{O3} = A1\ A0$



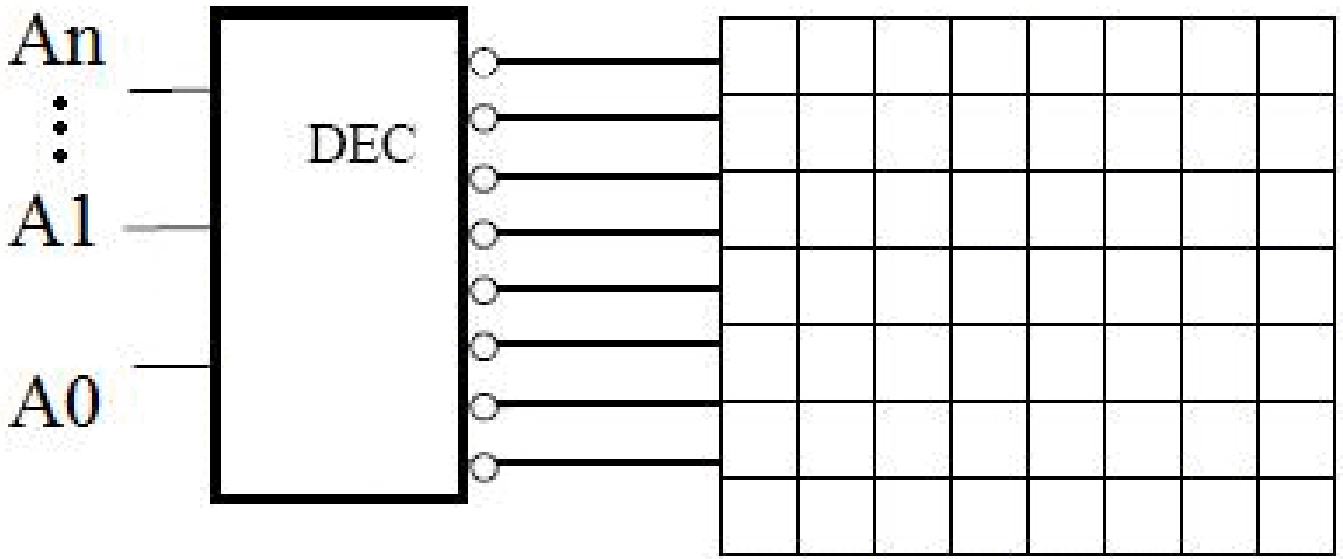
Decodificadores

Aplicaciones de un decodificador

- Un decodificador nos permite que en base a pocas entradas seamos capaces de controlar un gran número de salidas.

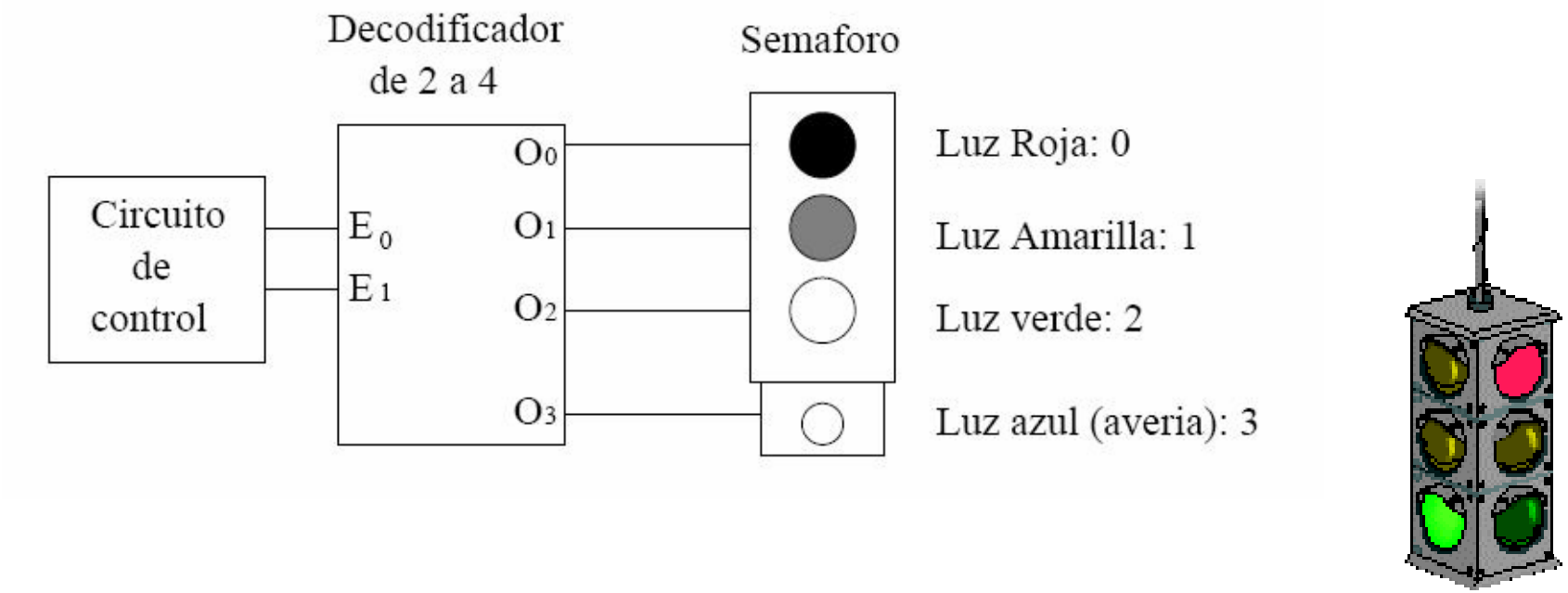
- Una de sus funciones principales es la de direccionar espacios de memoria ya que con N entradas puede direccionar hasta 2^N espacios de memoria.
- Para poder direccionar 1kb de memoria necesitaría 10 bits, ya que la cantidad de salidas seria 2^{10}, igual a 1024. Es decir, en lugar de distribuir una única entrada a una única salida, podemos simplificarlo de una forma particularmente sencilla utilizando para ello un decodificador ya que con N entradas puede direccionar hasta 2^N espacios de memoria.
- De esta manera: Con 20 bits => 2^{20} = 1Mb; Con 30 bits => 2^{30} = 1Gb, etc.

Decodificadores



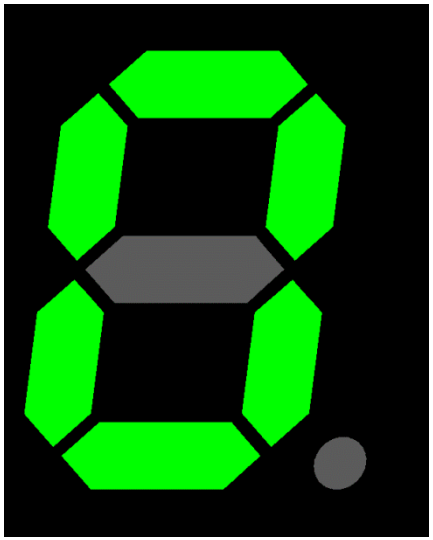
Decodificadores

- Un ejemplo particular del uso de un DEC 2 a 4 es un semáforo, ya que con 2 entradas controlamos 4 salidas.



Decodificadores

- Un tipo de decodificador muy empleado también con un DEC 2 a 4, es el de siete segmentos. Este circuito decodifica la información de entrada en BCD a un código de siete segmentos adecuado para que se muestre en un visualizador de siete segmentos, es decir un display.



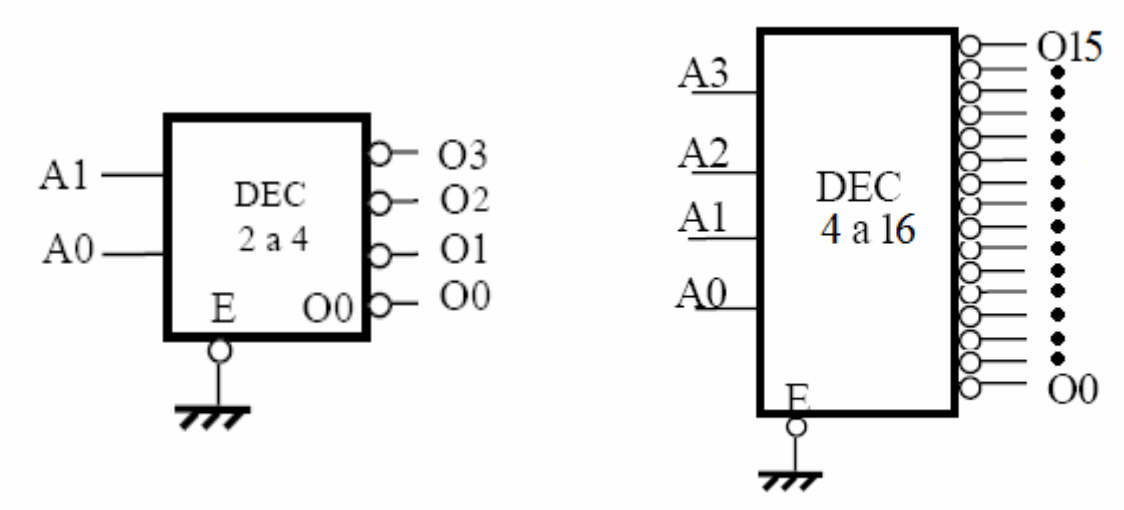
Decodificadores

Datos

- Un decodificador generalmente viene controlado por un habilitador (Enable) que suele estar conectado a tierra (“cero” lógico) para que este funcione siempre. Este tipo de control lo veremos usar en demultiplexores, ya que el demultiplexor y el decodificador con Enable se realizan con el mismo circuito.
- A partir de decodificadores “pequeños” podemos realizar conexiones para diseñar decodificadores de mayor tamaño.

Decodificadores

- Diseñar un decodificador de 4 entradas y 16 salidas en base a decodificadores de 2 entradas y 4 salidas (Desarrollo de 4 a 16 DEC en base a 2 a 4 DEC)



Decodificadores

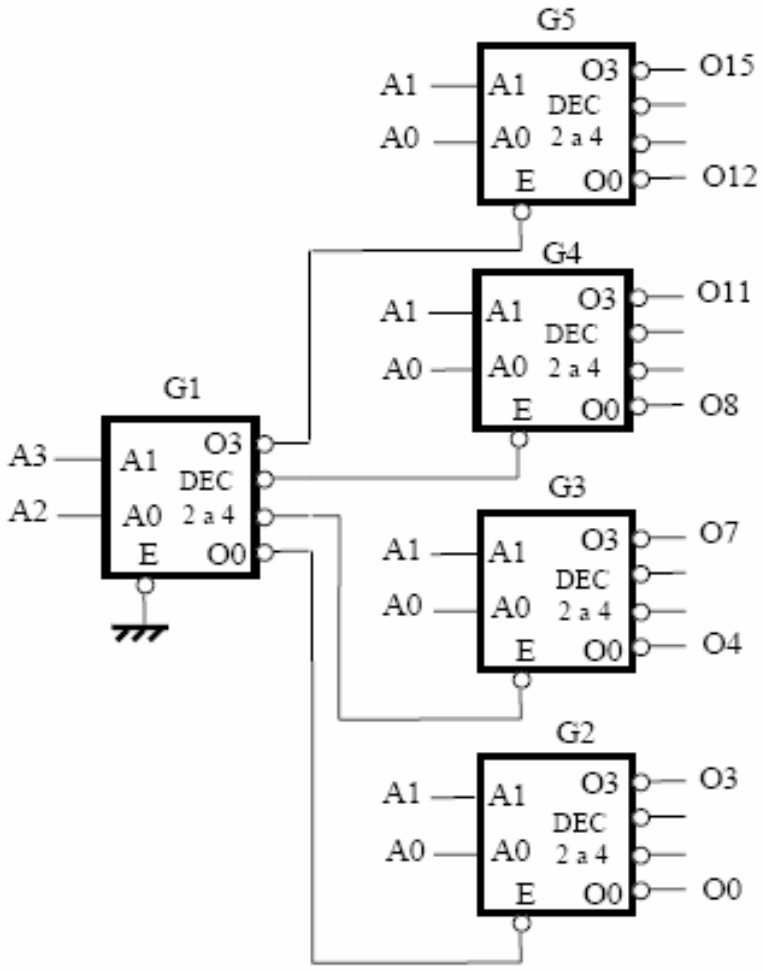
Desarrollo de 4 a 16 DEC en base a 2 a 4 DEC.

Si (A3A2) = 00, O0 de G1 es 1 (L) y G2 está habilitado y en él, en función de (A1A0) se obtienen las salidas O3-O0. G3, G4, G5 están deshabilitados: sus salidas son 0.

Si (A3A2) = 01, O1 de G1 es 1 (L) y G3 está habilitado y en él, en función de (A1A0) se obtienen las salidas O7-O4. G1, G4, G5 están deshabilitados: sus salidas son 0.

Si (A3A2) = 10, O2 de G1 es 1 (L) y G4 está habilitado y en él en función de (A1A0) se obtienen las salidas O11-O8. G1, G3, G5 están deshabilitados: sus salidas son 0.

Si (A3A2) = 11, O3 de G1 es 1 (L) y G5 está habilitado y en él en función de (A1A0) se obtienen las salidas O15-O12. G2, G3, G4 están deshabilitados: sus salidas son 0.



Decodificadores

- Modelo VDHL de un DEC 2 a 4

```
library ieee;
use.ieee.std_logic_1164.all;

entity dec2to4 is
port (A: in std_logic_vector(1 downto 0);      -- Entradas de dirección
      O: out std_logic_vector(3 downto 0));  -- Salidas
end dec2to4;
```

```
architecture DEC of dec2to4 is
begin
process (A)
   begin
      case A is
        when "00" => O <= "0001";
        when "01" => O <= "0010";
        when "10" => O <= "0100";
        when "11" => O <= "1000";
        when others => O <= "0000";
      end case;
end process;
end DEC;
```

Demultiplexores

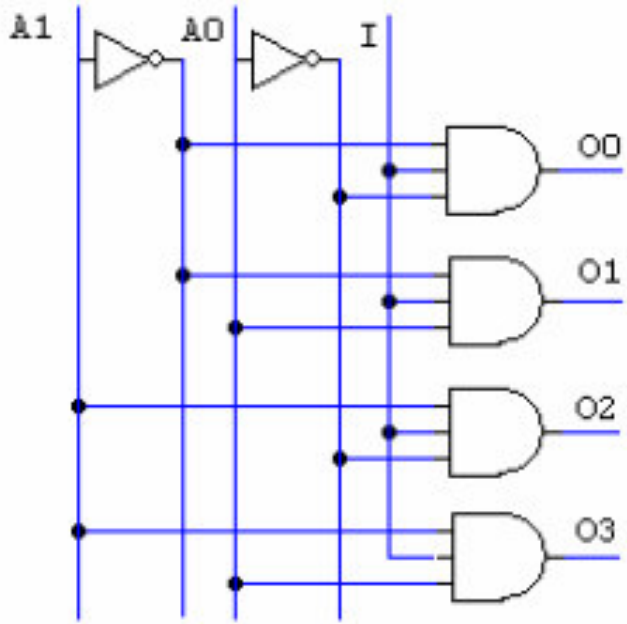
- Un demultiplexor es un circuito que pasa el valor lógico de una entrada I a una de sus M salidas Oj. Para determinar la salida a la que se envía la entrada I se usan N entradas de selección que sirven para indicar el índice j de la salida Oj a la que se envía la entrada I. Para N bits de dirección el número de salidas posibles donde enviar I es M = 2N.

1 de 4 DEMUX

A1	A0	O0	O1	O2	O3
0	0	I	0	0	0
0	1	0	I	0	0
1	0	0	0	I	0
1	1	0	0	0	I

$O0 = \overline{A1}\ \overline{A0}\ I$ $O1 = \overline{A1}\ A0\ I$

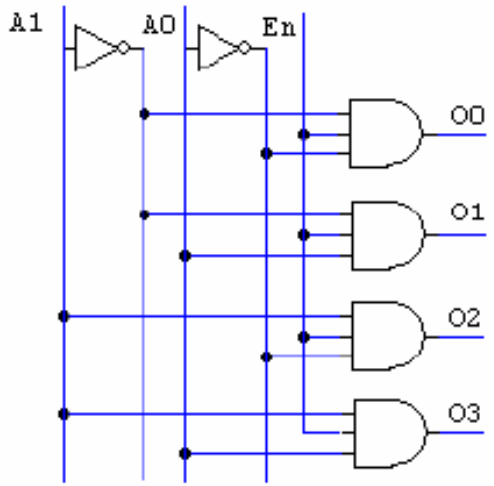
$O2 = A1\ \overline{A0}\ I$ $O3 = A1\ A0\ I$



Demultiplexores

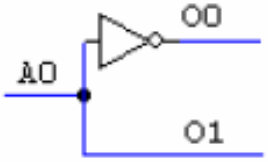
- La entrada de datos de un demultiplexor corresponde a un entrada de habilitación de un decodificador. Por tanto, el demultiplexor y el decodificador con Enable se realizan con el mismo circuito.

E	A1	A0	O0	O1	O2	O3
0	X	X	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1

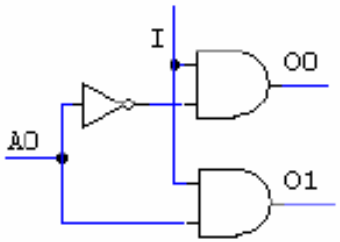


- Un decodificador 1 a 2 sin habilitador puede realizarse sólo con un inversor, con habilitador (o un demultiplexor 1 de 2) no.

A0	O0	O1
0	1	0
1	0	1



A0	O0	O1
0	I	0
1	0	I

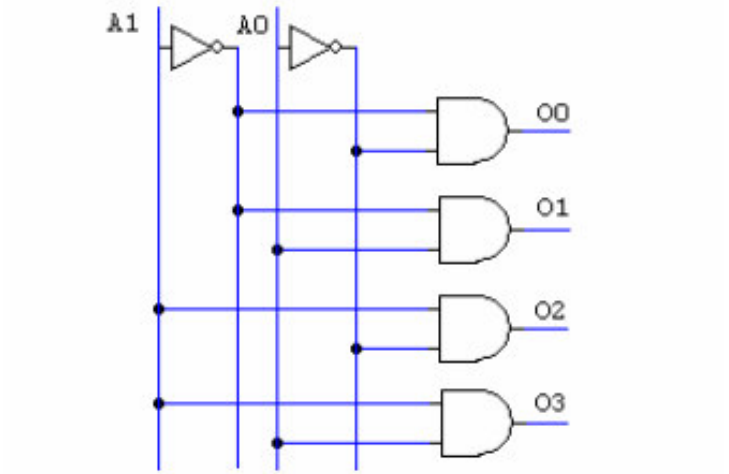


Demultiplexores

Como vemos, el decodificador funciona como un demultiplexor si contamos el Enable como una entrada más

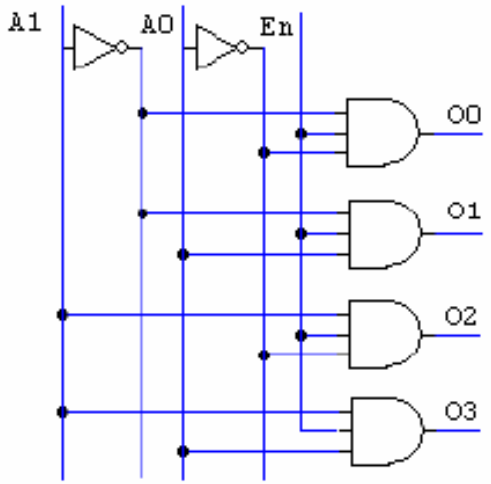
2 a 4 DEC

A1	A0	O0	O1	O2	O3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1



1 de 4 DEMUX

E	A1	A0	O0	O1	O2	O3
0	X	X	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1



Demultiplexores

Modelo VHDL de un 2 a 4 DEC (ó 1 de 4 DEMUX)

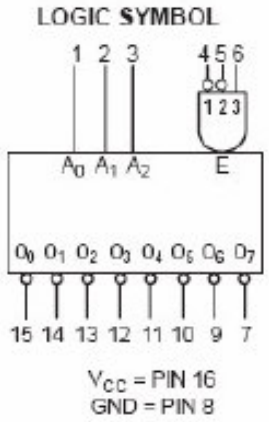
```
library ieee;
use.ieee.std_logic_1164.all;

entity dec2to4 is
port (A: in std_logic_vector(1 downto 0);      -- Entradas de dirección
      E: in std_logic;                         -- Entrada de habilitación
      O: out std_logic_vector(3 downto 0));    -- Salidas
end dec2to4;
```

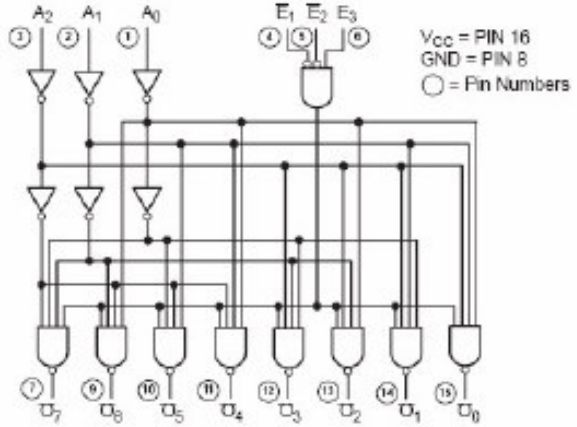
```
architecture DEC of dec2to4 is
begin
process (A, E)
    begin
    if E = '0' then
       O <= "0000";
    else
        case A is
          when "00" => O <= "0001";
          when "01" => O <= "0010";
          when "10" => O <= "0100";
          when "11" => O <= "1000";
          when others => O <= "0000";
        end case;
     end if;
end process;
end DEC;
```

```
architecture DEMUX of dec2to4 is
begin
process (A, E)
    begin
    case A is
        when "00" => O(0) <= E; O(1) <= '0';
                     O(2) <= '0'; O(3) <= '0';
        when "01" => O(0) <= '0'; O(1) <= E;
                     O(2) <= '0'; O(3) <= '0';
        when "10" => O(0) <= '0'; O(1) <= '0';
                     O(2) <= E; O(3) <= '0';
        when "11" => O(0) <= '0'; O(1) <= '0';
                     O(2) <= '0'; O(3) <= E;
        when others => O(0) <= '0'; O(1) <= '0';
                       O(2) <= '0'; O(3) <= '0';
     end case;
end process;
end DEMUX;
```

Decodificadores/demultiplexores

Circuitos comerciales: 74'138: un 3 a 8 DEC/ 1 de 8 DEMUX.
Las salidas de los decodificadores están en polaridad negativa.
3 entradas de habilitación, 2 de polaridad negativa y 1 positiva.





TRUTH TABLE

INPUTS						OUTPUTS							
$\overline{E}_1$	$\overline{E}_2$	E_3	A_0	A_1	A_2	$\overline{O}_0$	$\overline{O}_1$	$\overline{O}_2$	$\overline{O}_3$	$\overline{O}_4$	$\overline{O}_5$	$\overline{O}_6$	$\overline{O}_7$
H	X	X	X	X	X	H	H	H	H	H	H	H	H
X	H	X	X	X	X	H	H	H	H	H	H	H	H
X	X	L	X	X	X	H	H	H	H	H	H	H	H
L	L	H	L	L	L	L	H	H	H	H	H	H	H
L	L	H	H	L	L	H	L	H	H	H	H	H	H
L	L	H	L	H	L	H	H	L	H	H	H	H	H
L	L	H	H	H	L	H	H	H	L	H	H	H	H
L	L	H	L	L	H	H	H	H	H	L	H	H	H
L	L	H	H	L	H	H	H	H	H	H	L	H	H
L	L	H	L	H	H	H	H	H	H	H	H	L	H
L	L	H	H	H	H	H	H	H	H	H	H	H	L

H = HIGH Voltage Level
L = LOW Voltage Level
X = Don't Care

Problemas propuestos

- Se quiere diseñar un decodificador de 40 direcciones de 0 a 39 utilizando decodificadores binarios (2 a 4, 3 a 8, 4 a 16, etc). Indicar cuál es el número mínimo de decodificadores binarios que hay que utilizar y realizar el diseño del decodificador utilizando los decodificadores binarios y las puertas lógicas que sean necesarias (un inversor).

- Diseñar un multiplexor de 16 entradas utilizando 4 multiplexores triestado de 4 entradas con habilitador (el circuito deshabilitado queda en alta impedancia) y un decodificador 2 a 4.