Multiplexores

- También llamado, selector de datos, un multiplexor es un circuito combinacional que selecciona una de ***n*** líneas de entrada (donde n es un numero entero positivo) y transmite su información binaria a la salida.

- La selección de la entrada es controlada por un conjunto de líneas de selección o control.

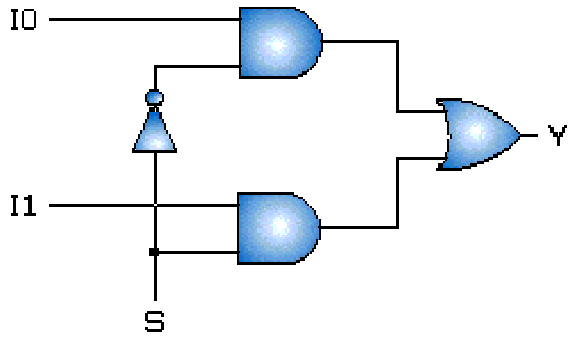
- La relación de líneas de entrada y líneas de selección está dada por la expresión ***2^n***, donde ***n*** corresponde al número de líneas de selección y ***2^n*** al número de líneas de entrada.

Multiplexores

- **Multiplexor 2 a 1**

  -Las entradas son *I0* e *I1*
  -La selección viene dada por el valor de la entrada S
  -La salida será igual a:

$$Y=\overline{S}I0+SI1$$



Multiplexor 2 a 1

S	Y
0	I_0
1	I_1

Tabla de verdad

Multiplexores

- **Multiplexor 4 a 1**

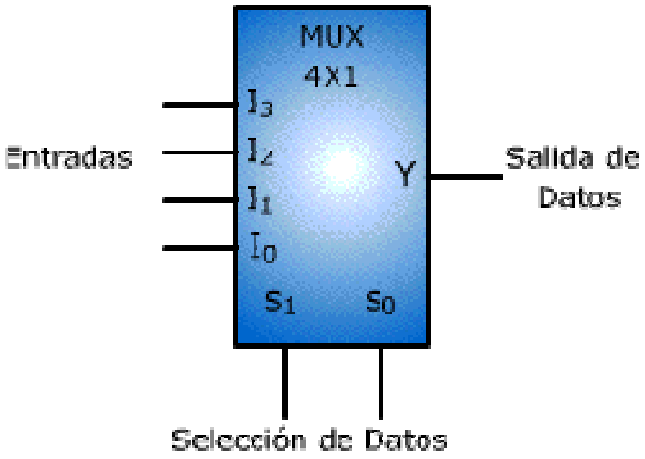
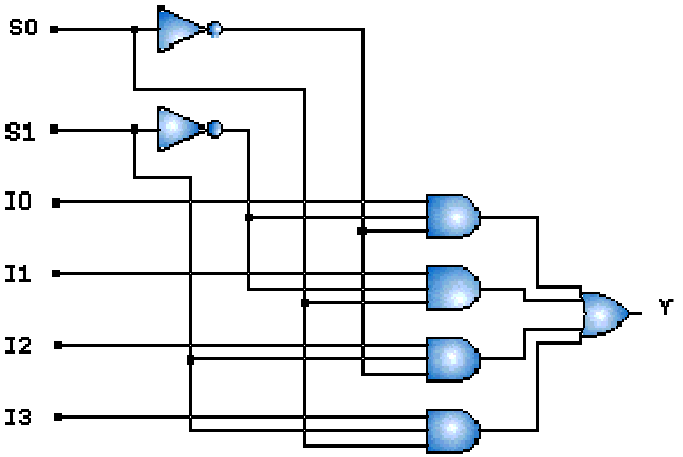


Diagrama de bloques



Circuito lógico

La salida será igual a:

$$Y = I0 \cdot \overline{S1} \cdot \overline{S0} + I1 \cdot \overline{S1} \cdot S0 + I2 \cdot S1 \cdot \overline{S0} + I3 \cdot S1 \cdot S0$$

Multiplexores

- **Descripción en VHDL**

  Un multiplexor representa en un circuito digital una situación de múltiple decisión correspondiente a una sentencia **if-else** en VHDL (2 entradas) o una sentencia **case** para *N* entradas. Ejemplo de multiplexor de dos entradas:

```
entity MUX2a1 is
   port(a: in std_logic;
   b: in std_logic;
   sel: in std_logic;
   z: out std_logic);
end entity

architecture dataflow of MUX2a1 is
   begin
     z <= a when sel='0' else b;
end dataflow;
```

Multiplexores

Ejemplo de multiplexor de cuatro entradas. Este ejemplo trabaja con vectores para controlarla entrada activa a través de la entrada *sel.*:

```
entity MUX4a1 is
    port(
    a: in std_logic;
    b: in std_logic;
    c: in std_logic;
    d: in std_logic;
    z: out std_logic;
    sel: in std_logic_vector(1 downto 0));
end entity;
```

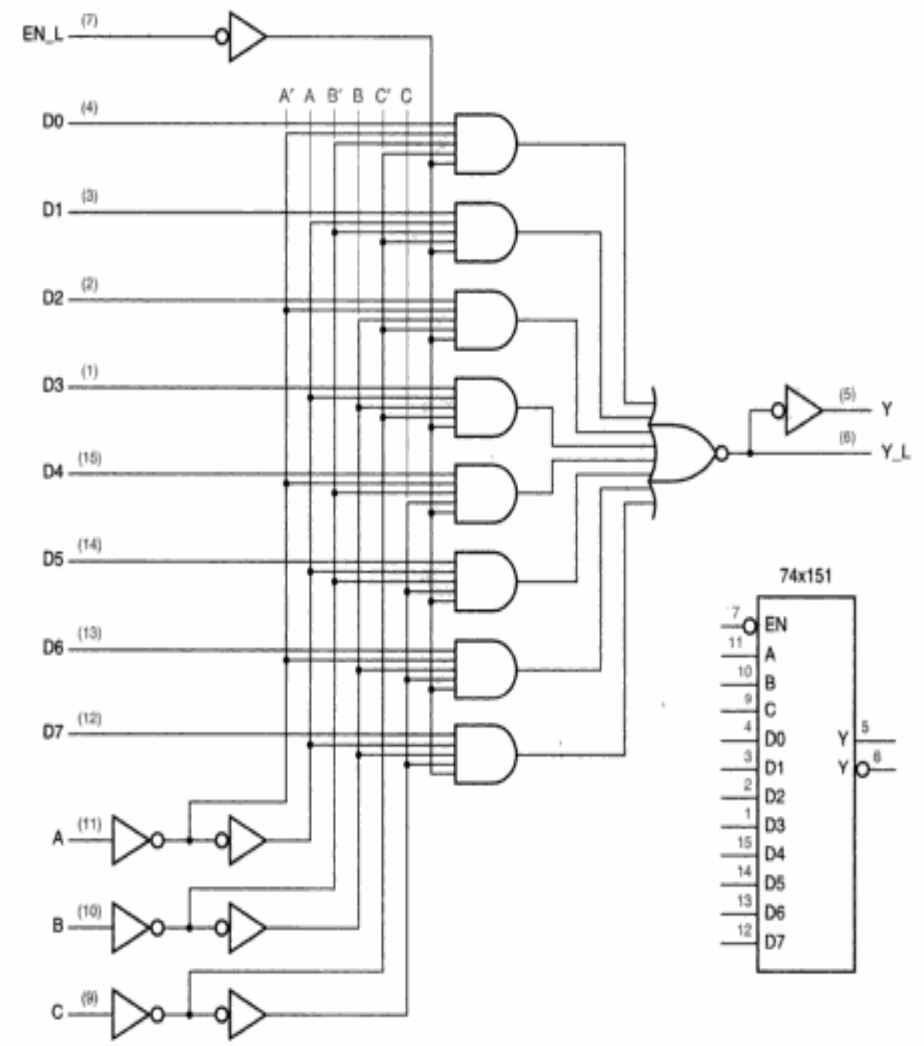
```
architecture dataflow of MUX4a1 is
  begin
  process(a,b,c,d,sel) begin
    case sel is
          when "00" => z <= a;
          when "01" => z <= b;
          when "10" => z <= c;
          when "11" => z <= d;
          when others => z <= '-';
     end case;
  end process;
end dataflow;
```

Ejemplos comerciales

- 74x151

Todos los multiplexores que se usan comúnmente vienen en encapsulados de 16 terminales.

El 74x151 tiene 8 entradas de 1 bit. En el esquema serian las entradas de D0 a D8. Las entradas de selección seria las A, B, C. La entrada EN_L es activa baja y se suministran salidas en activa alta y baja, que serian Y e Y_L.



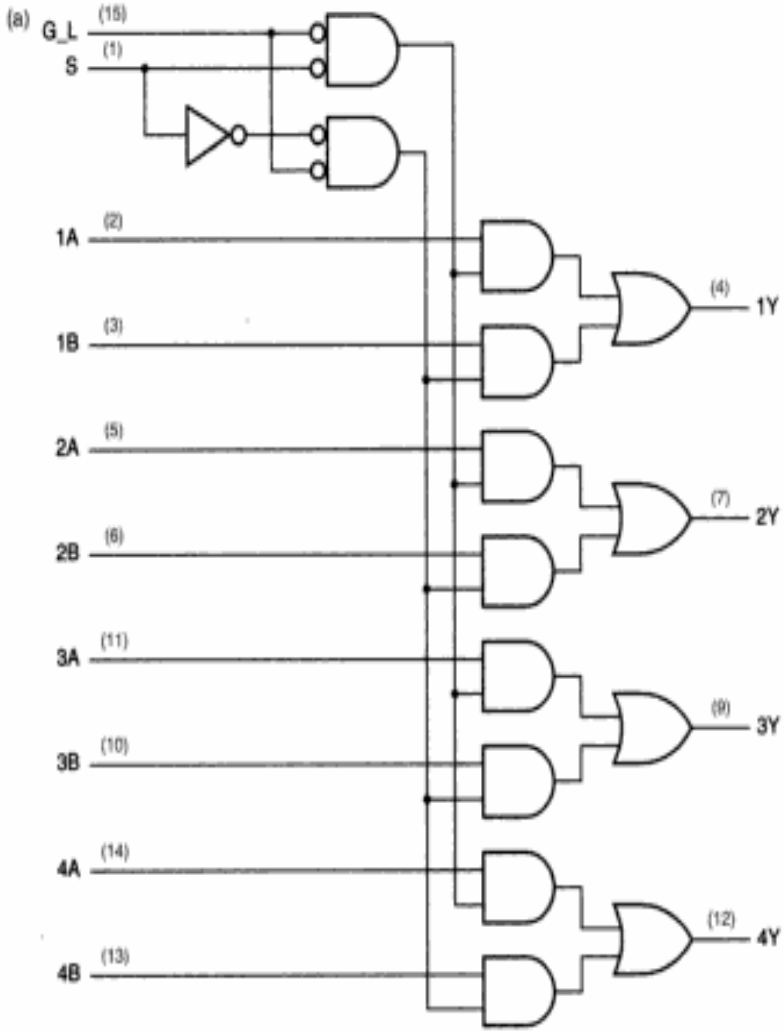
La tabla de la verdad sería esta:

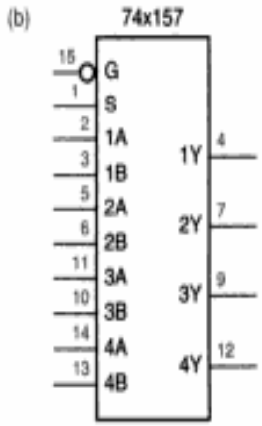
Entradas				Salidas	
EN_L	C	B	A	Y	Y_L
1	x	x	x	0	1
0	0	0	0	D0	D0′
0	0	0	1	D1	D1′
0	0	1	0	D2	D2′
0	0	1	1	D3	D3′
0	1	0	0	D4	D4′
0	1	0	1	D5	D5′
0	1	1	0	D6	D6′
0	1	1	1	D7	D7′

Ejemplos comerciales

- 74x157

El 74x157 tiene 2 entradas de 4 bit. Las fuentes de datos corresponderían a A y B, y las entradas de un bit serían de A1 a A4, y de B1 a B4. La entrada de selección seria S, y G_L la de habilitación de activa baja.





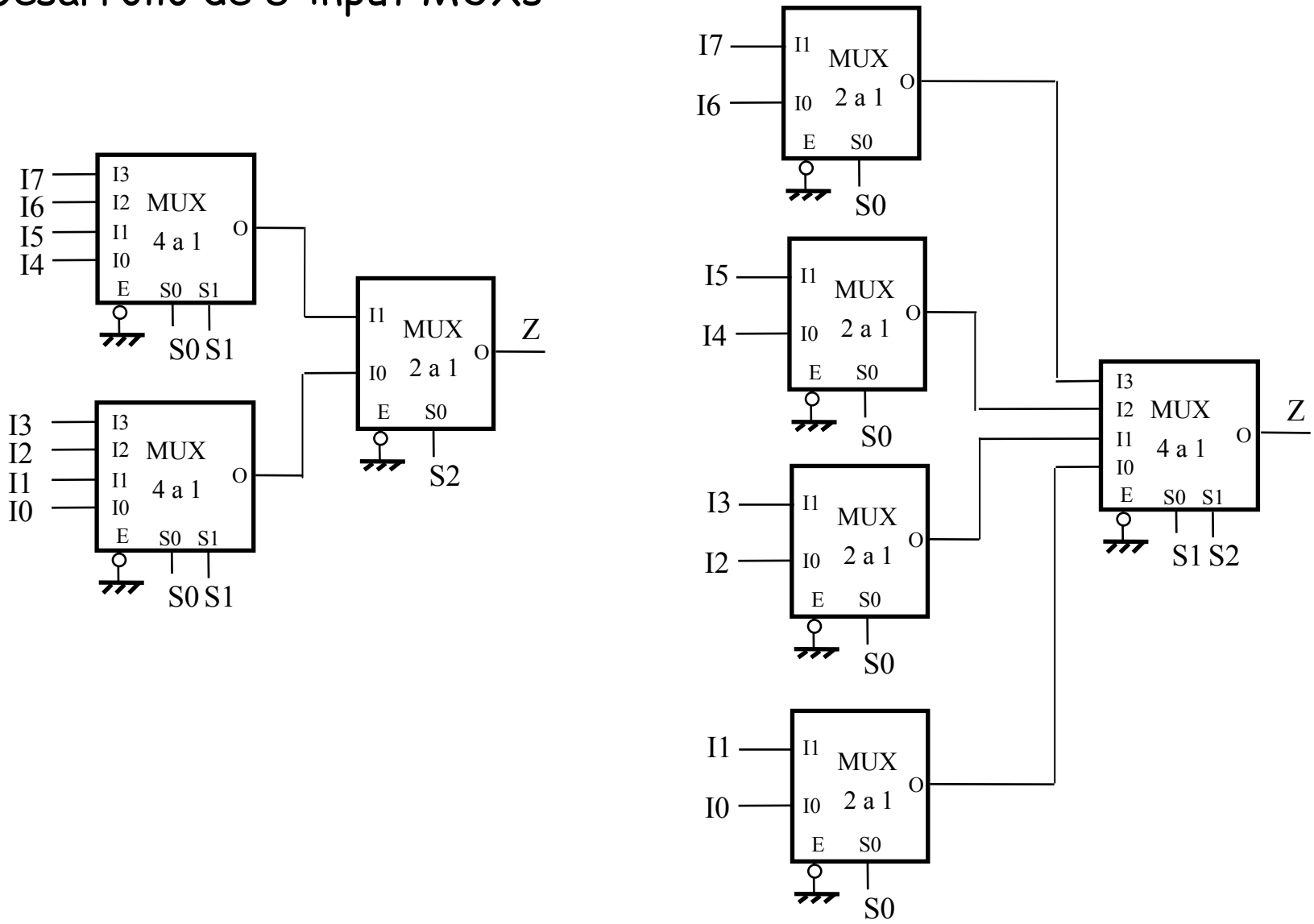
La tabla de la verdad sería esta:

Entradas		Salidas			
G_L	S	1Y	2Y	3Y	4Y
1	x	0	0	0	0
0	0	1A	2A	3A	4A
0	1	1B	2B	3B	4B

Multiplexores

Desarrollo de N-input MUXs en base a M-input MUX (N > M)

Desarrollo de 8-input MUXs



Multiplexores

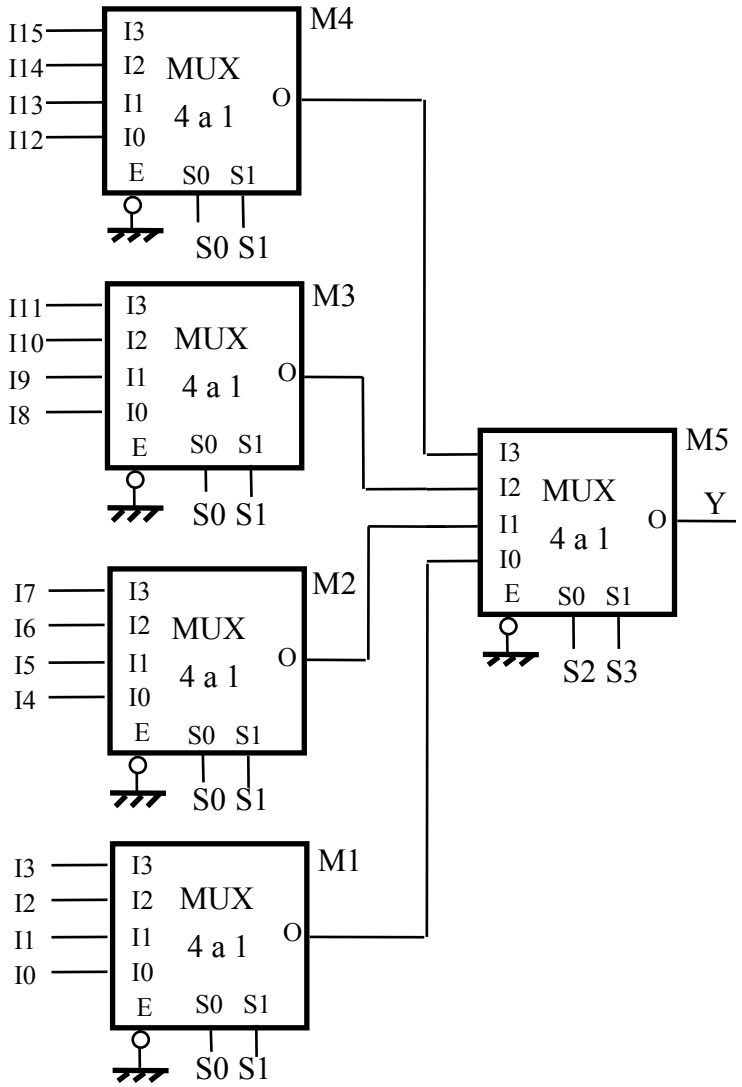
Desarrollo de 16-input MUXs en base a 4-Input MUXs

Si (S3S2) = 00, Y <= I0 de M5 que está conectada a la salida de M1. En función de (S1S0) se selecciona en M1 las entradas I3-I0.

Si (S3S2) = 01, Y <= I1 de M5 que está conectada a la salida de M2. En función de (S1S0) se selecciona en M2 las entradas I4-I7.

Si (S3S2) = 10, Y <= I2 de M5 que está conectada a la salida de M3. En función de (S1S0) se selecciona en M3 las entradas I8-I11.

Si (S3S2) = 11, Y <= I3 de M5 que está conectada a la salida de M4. En función de (S1S0) se selecciona en M4 las entradas I12-I15.



Multiplexores

- **Aplicaciones de los multiplexores**
  - **Realización de funciones lógicas:** Utilizando inversores y conectando a 0 o 1 las entradas según interese, se consigue diseñar funciones complejas, de un modo mas compacto que con las tradicionales puertas lógicas.

Problemas propuestos

- Construir un multiplexor de 5 entradas
  a) utilizando puertas lógicas.
  b) utilizando multiplexores de dos entradas.

Problemas propuestos

- Diseñar un circuito multiplexor con prioridad de 4 bits. El circuito tiene 4 entradas de datos (I3-I0), 4 entradas de selección (S3-S0) y dos salidas Z y G. Cuando una o más de las entradas S están a 1, Z toma el valor de la entrada Ii, siendo i es el índice más alto de las entradas Si que están a 1; si todas las entradas S3-S0 están a 0, entonces Z toma el valor 0. La salida G se fija a 1 si al menos alguna entrada Si está a 1, en caso contrario se fija a 0.

  a) Mostrar en una tabla el comportamiento lógico del circuito. Encontrar las ecuaciones lógicas de la salidas Z y G expresándolas en dos niveles y en forma factorizada.

  b) Implementar la expresión factorizada de Z utilizando multiplexores de 2 entradas.

  c) Realizar una descripción VHDL del multiplexor.

Problemas propuestos

- Un circuito de "desplazamiento en barril" ("barrel-shifter") mueve los datos de entrada de forma que aparezcan en la salida girados el número de posiciones marcados por las señales de control. Construir utilizando multiplexores un "barrel-shifter" de 4 bits de entrada (a3a2a1a0) y 4 bits de salida (z3z2z1z0) con 4 posibles desplazamientos (dos señales de control c1c0):

  (c1c0) = 0 => (z3z2z1z0) = (a3a2a1a0),

  (c1c0) = 1 => (z3z2z1z0) = (a2a1a0a3),

  (c1c0) = 2 => (z3z2z1z0) = (a1a0a3a2),

  (c1c0) = 3 => (z3z2z1z0) = (a0a3a2a1).