14

BÚSQUEDA TABÚ (EJEMPLO)

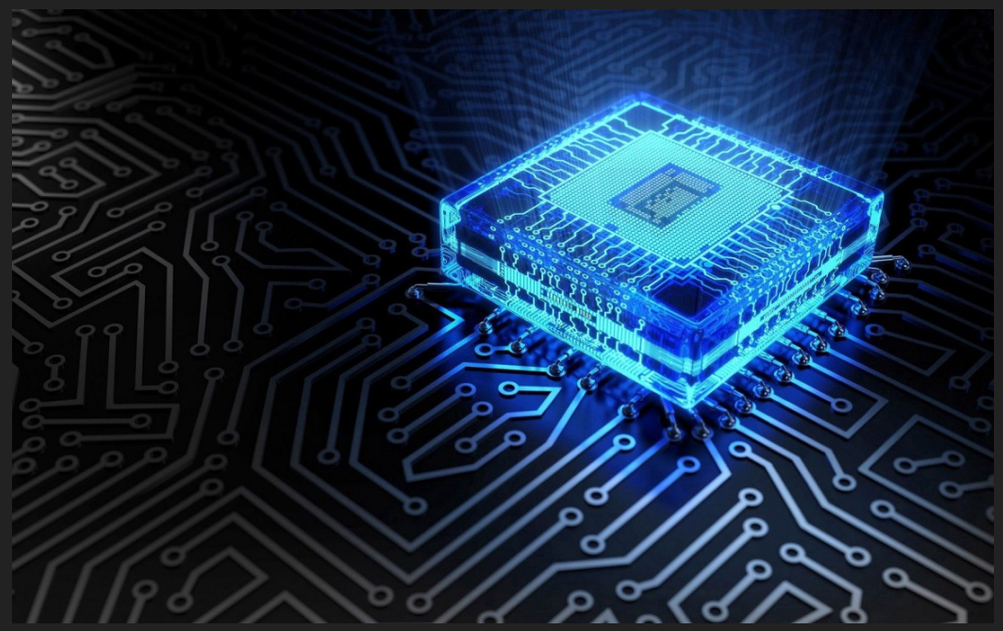
OPTIMIZACIÓN PPA

- En el diseño de circuitos integrados PPA significa Power, Performance y Area.
- Siempre se busca optimizar esas variables, es decir:

  ✓ Power (Potencia): Minimizar el consumo de energía.

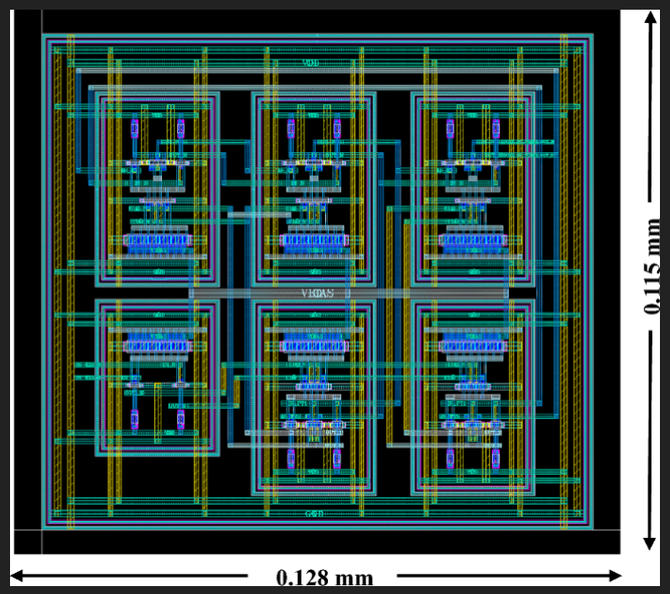
  ✓ Performance (Rendimiento): Maximizar el rendimiento.

  ✓ Area (Área): Minimizar el área física del circuito en el chip.

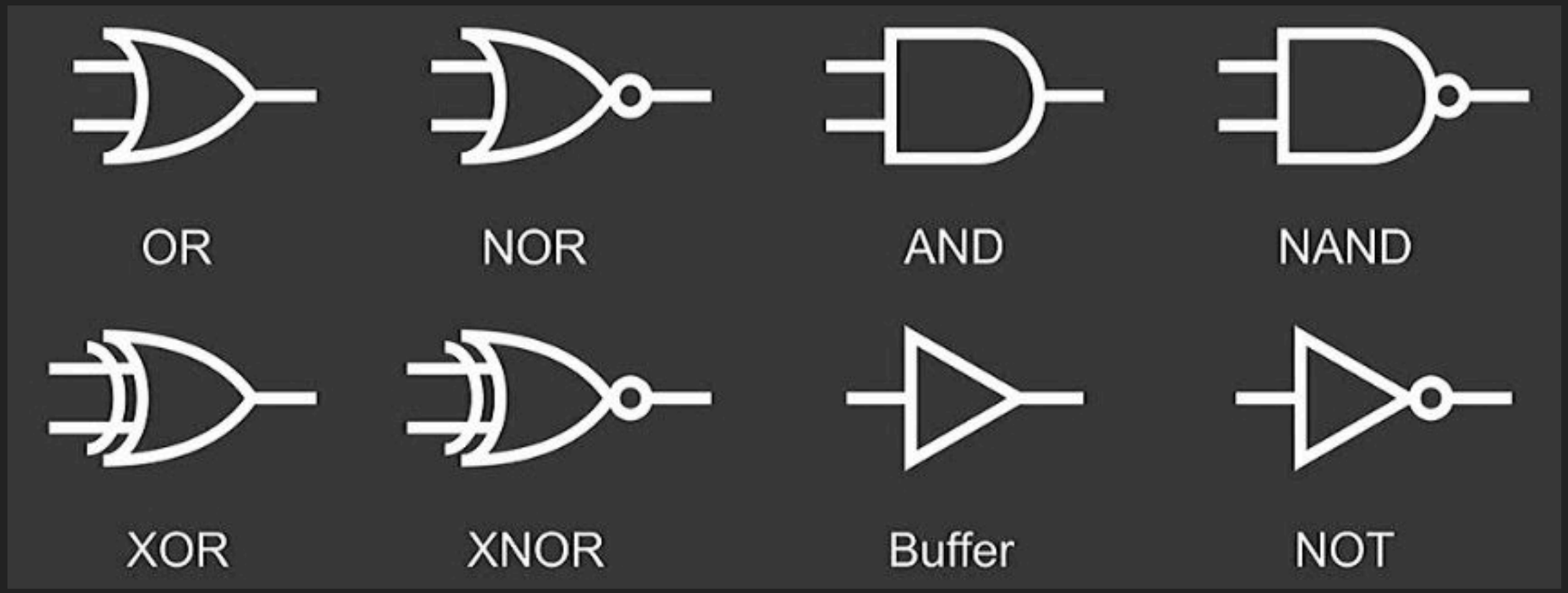


¿QUÉ IMPLICA MAXIMIZAR EL RENDIMIENTO?

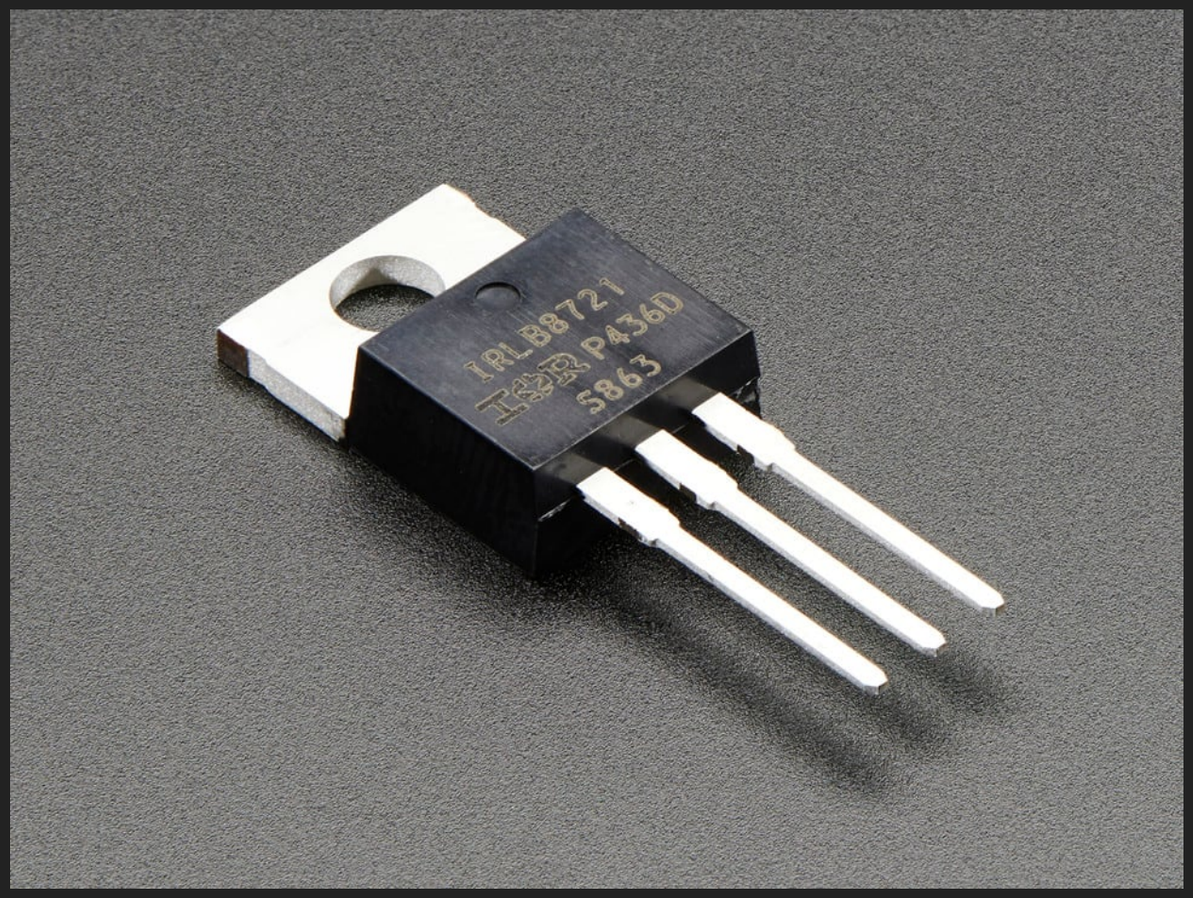
- Menor Retardo: Las señales eléctricas tienen una velocidad de propagación finita. Reducir las distancias de conexión puede disminuir el retardo de señal, lo que permite que las señales lleguen más rápido a su destino.
- Menor Interferencia: Con conexiones más cortas, hay menos riesgo de interferencias entre señales, lo que puede mejorar la integridad de la señal y reducir el ruido en el circuito.

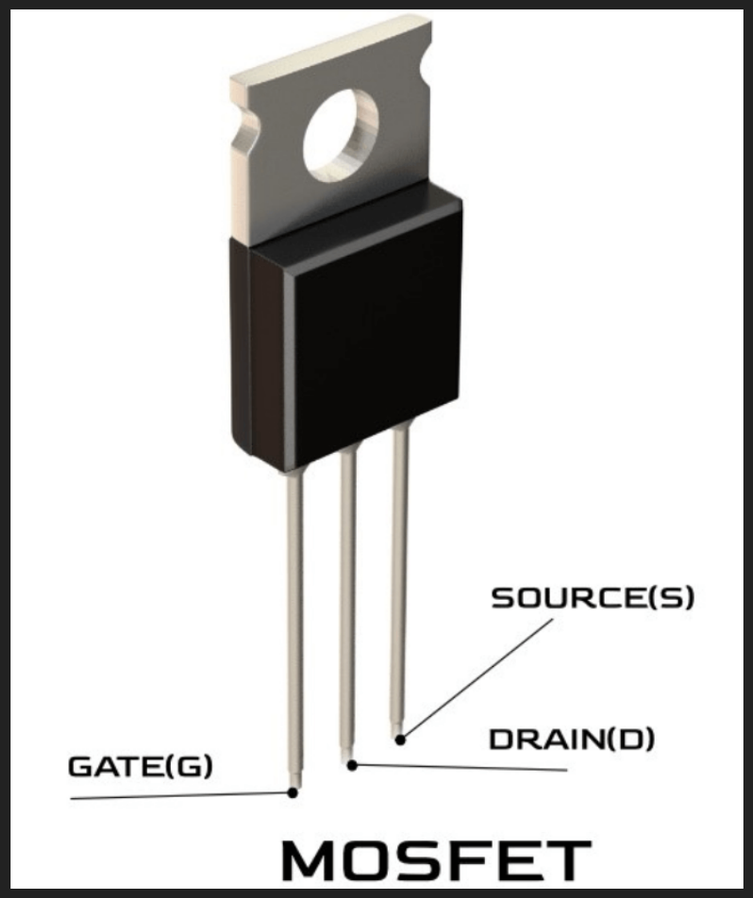


DISEÑO DE UNA COMPUERTA LÓGICA

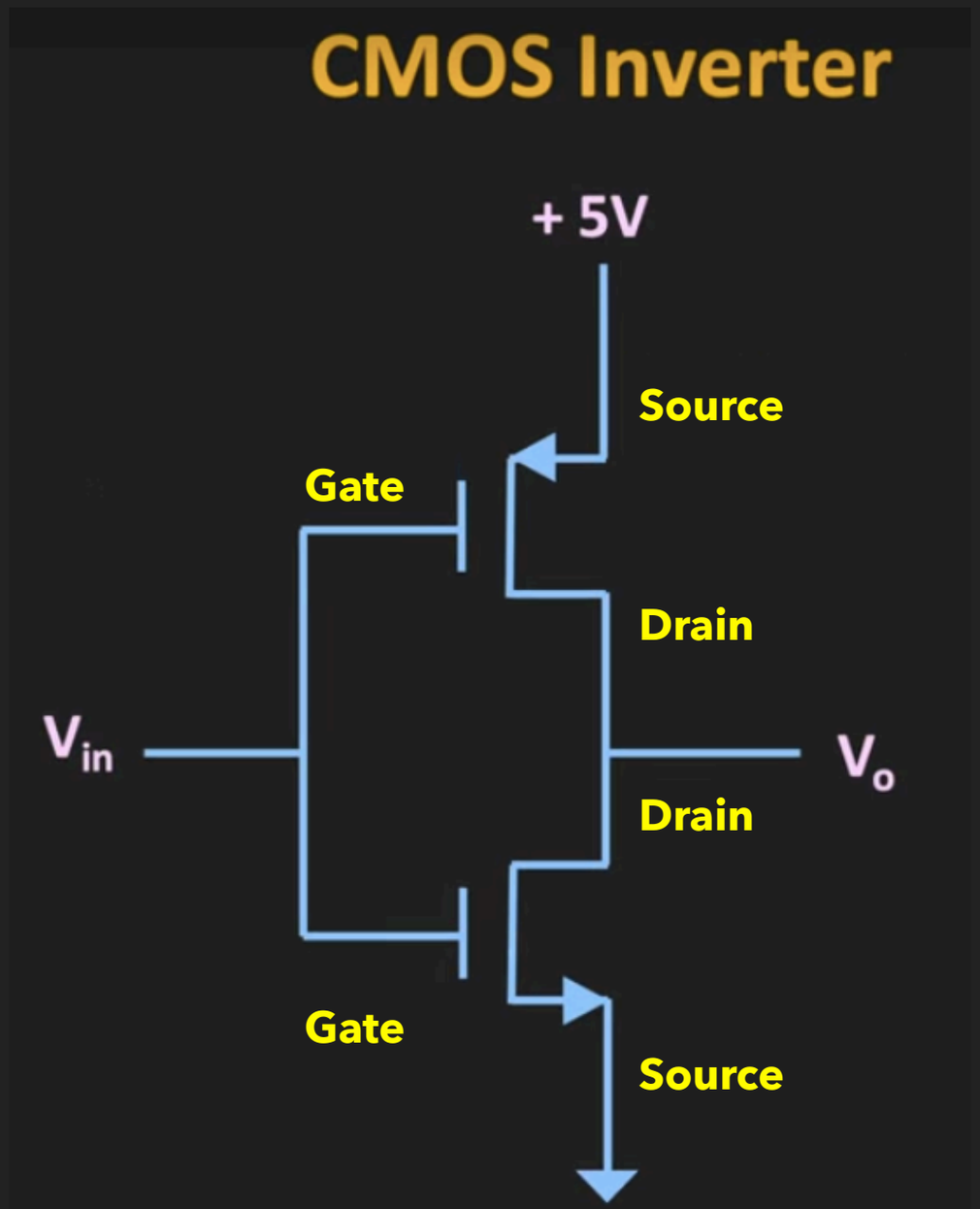


DISEÑO DE UNA COMPUERTA LÓGICA

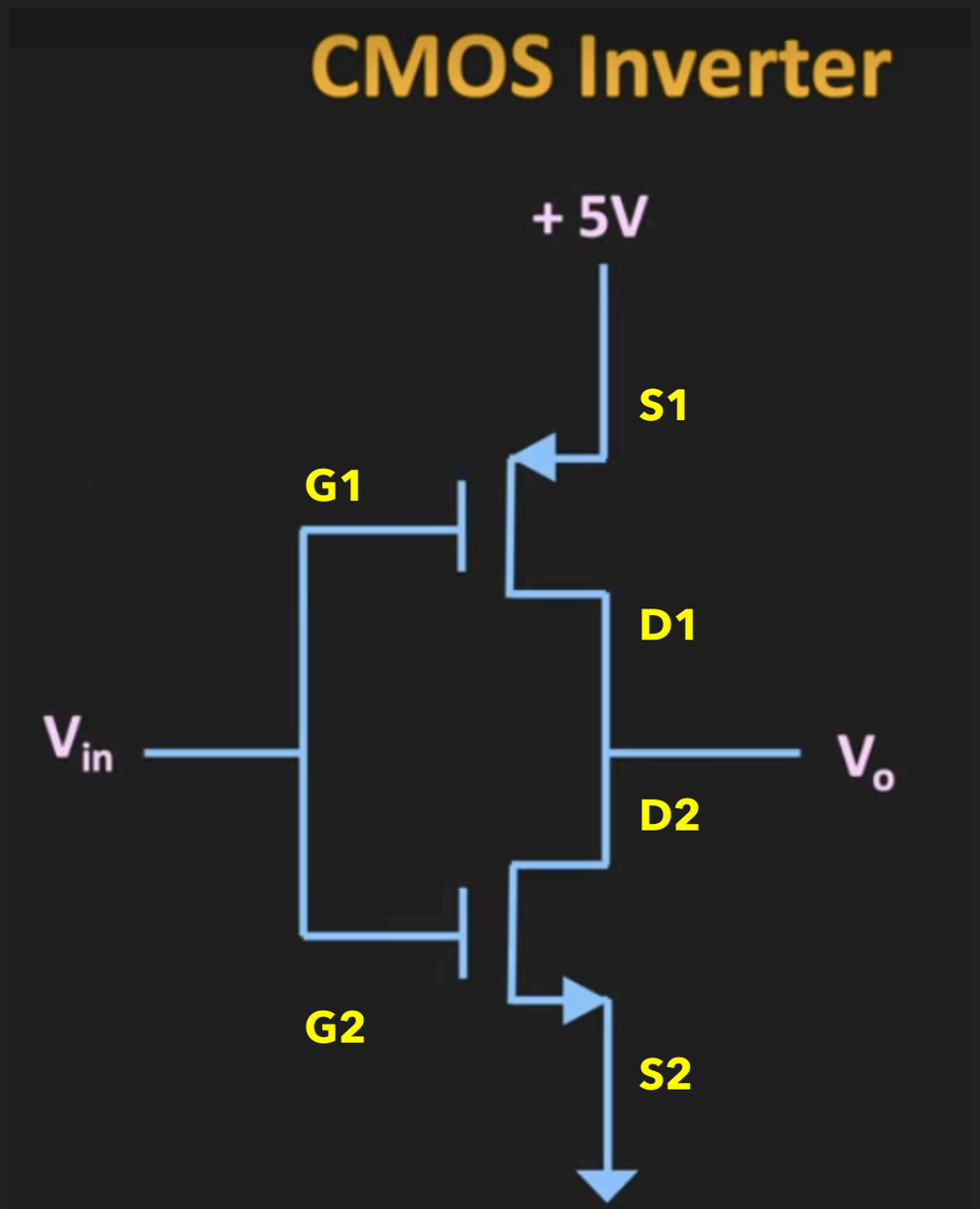




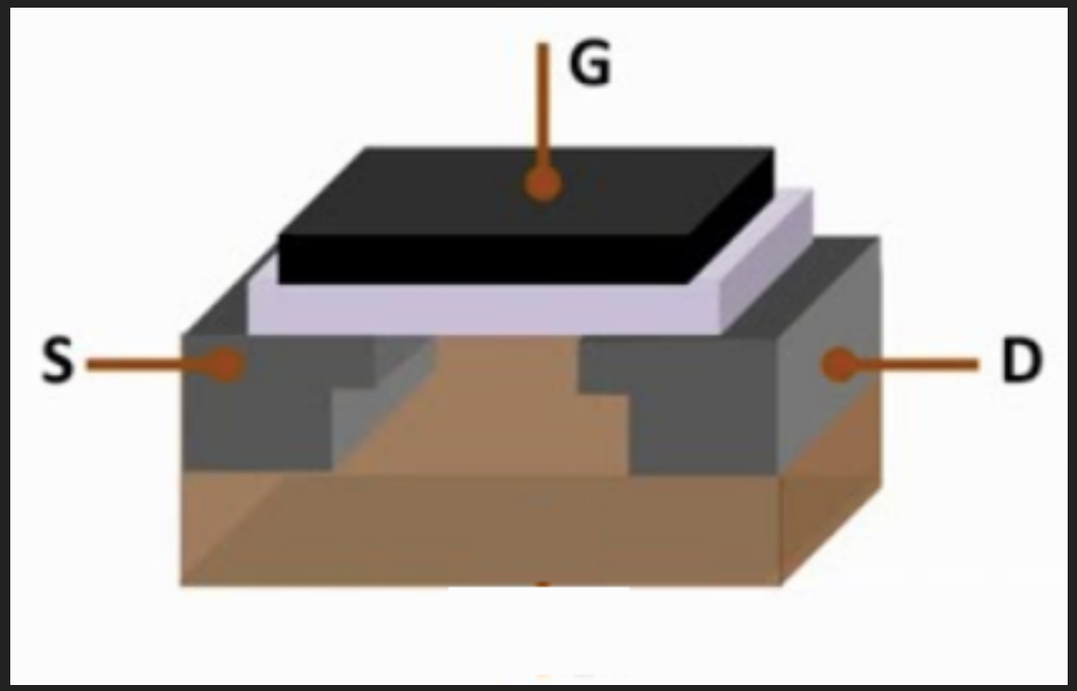
COMPUERTA NOT (CIRCUITO)



COMPUERTA NOT (CIRCUITO)



COMPUERTA NOT (ASPECTO FÍSICO HIPOTÉTICO)



¿CÓMO CONECTAR AMBOS TRANSISTORES MINIMIZANDO LA LONGITUD DE CONEXIÓN?

- Definir una matriz de conexiones:

	d1	s1	g1	d2	s2	g2
d1	0	0	0	1	0	0
s1	0	0	0	0	0	0
g1	0	0	0	0	0	1
d2	1	0	0	0	0	0
s2	0	0	0	0	0	0
g2	0	0	1	0	0	0

¿CÓMO CONECTAR AMBOS TRANSISTORES MINIMIZANDO LA LONGITUD DE CONEXIÓN?

- Definir una matriz de distancias:

$$d(Euc) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

```python
# ------------------------------------------------------------------
# se calcula la distancia euclidiana entre dos puntos (en 3D en este caso )
# la dist. euclid. es d=raiz_cuuadrada((x2-x1)^2+(y2-y1)^2+(z2-z1)^2)
# ------------------------------------------------------------------
def distancia(punto1, punto2):
    return np.linalg.norm(np.array(punto1) - np.array(punto2))


# ------------------------------------------------------------------
# generacion de la matriz de distancias entre los terminales
# ------------------------------------------------------------------
def generar_matriz_distancias(posiciones_terminales):
    num_terminales = len(posiciones_terminales)
    matriz_distancias = np.zeros((num_terminales, num_terminales))
    for i in range(num_terminales):
        for j in range(i + 1, num_terminales):
            matriz_distancias[i][j] = distancia(posiciones_terminales[i], posiciones_terminales[j])
            matriz_distancias[j][i] = matriz_distancias[i][j]  # simetria
    return matriz_distancias
```

¿CÓMO CONECTAR AMBOS TRANSISTORES MINIMIZANDO LA LONGITUD DE CONEXIÓN?

- Veamos el código fuente…

REFERENCIAS BIBLIOGRÁFICAS Y WEB (III)

- Glover and Laguna, Tabu search in Pardalos and Resende (eds.), Handbook of Applied Optimization, Oxford Academic Press, 2002.
- CMOS Logic Gates Explained | Logic Gate Implementation using CMOS logic https://www.youtube.com/watch?v=f3zRz0d9XA8