

# IMPLEMENTACIÓN NAND-NOR

*Técnicas Digitales I*

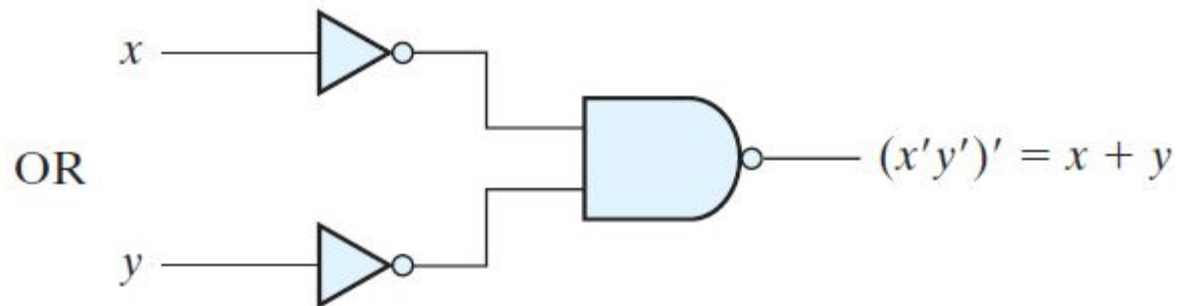
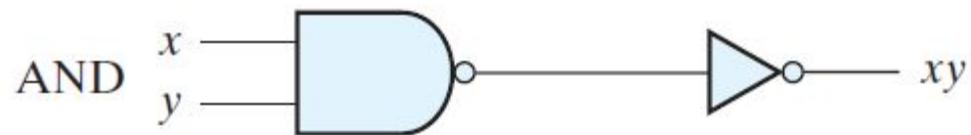
---

Luis Eduardo Toledo

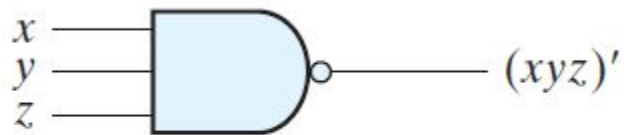


# OPERACIONES LÓGICAS CON COMPUERTAS

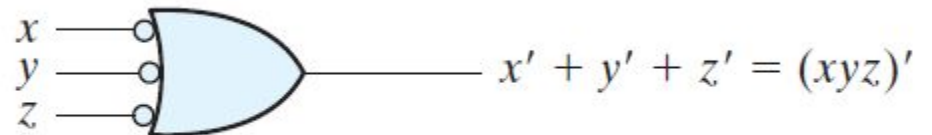
## NAND



# DOS SÍMBOLOS LÓGICOS PARA LA COMPUERTA NAND

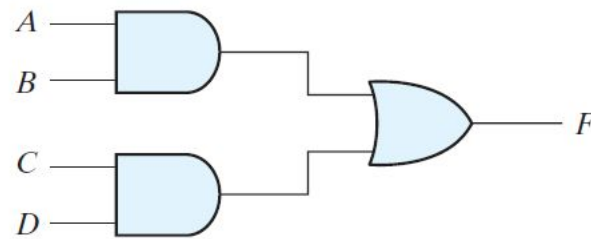


(a) AND-invert

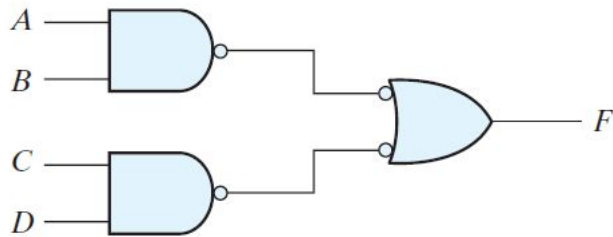


(b) Invert-OR

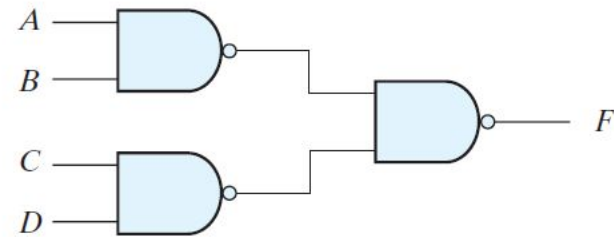
# TRES FORMAS PARA IMPLEMENTAR LA FUNCIÓN $F = A.B + C.D$



(a)



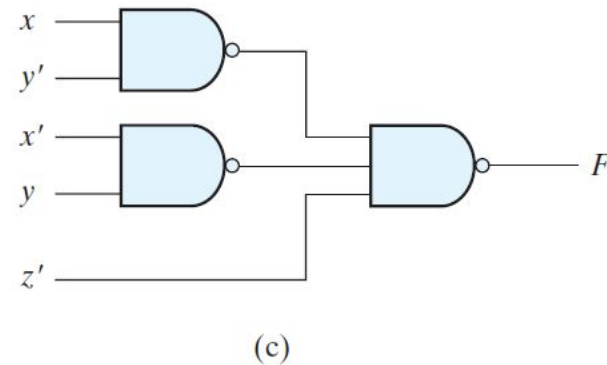
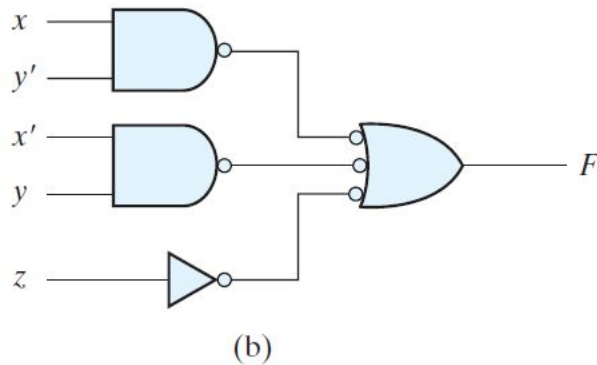
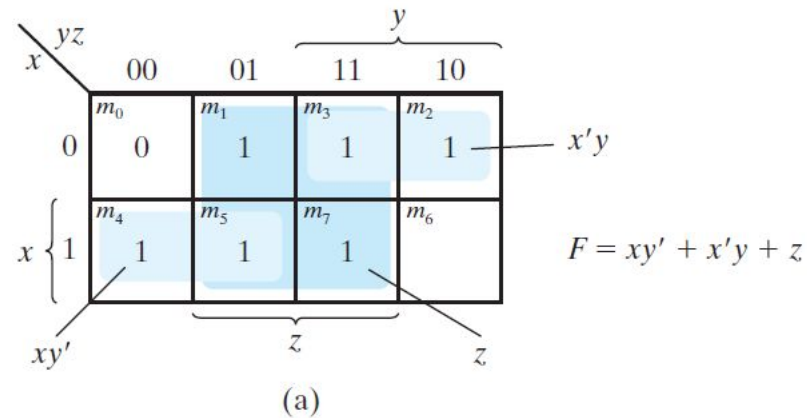
(b)



(c)

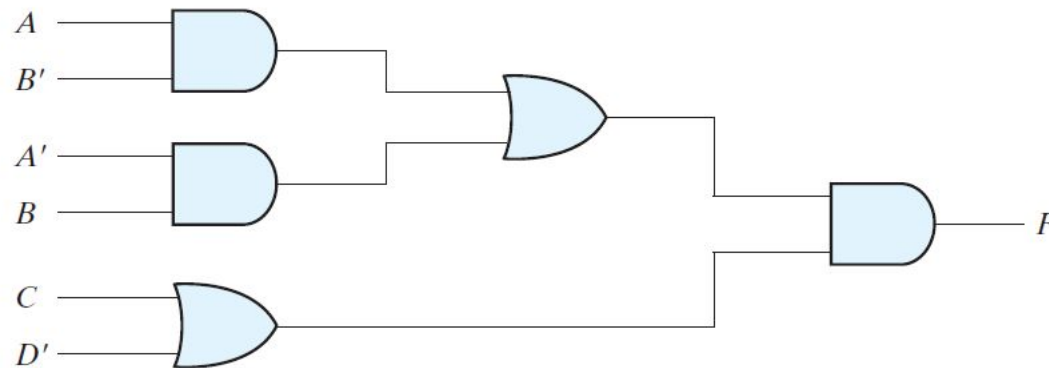
# EJEMPLO DE IMPLEMENTACIÓN DE UNA FUNCIÓN USANDO NAND

$$F(x, y, z) = (1, 2, 3, 4, 5, 7)$$

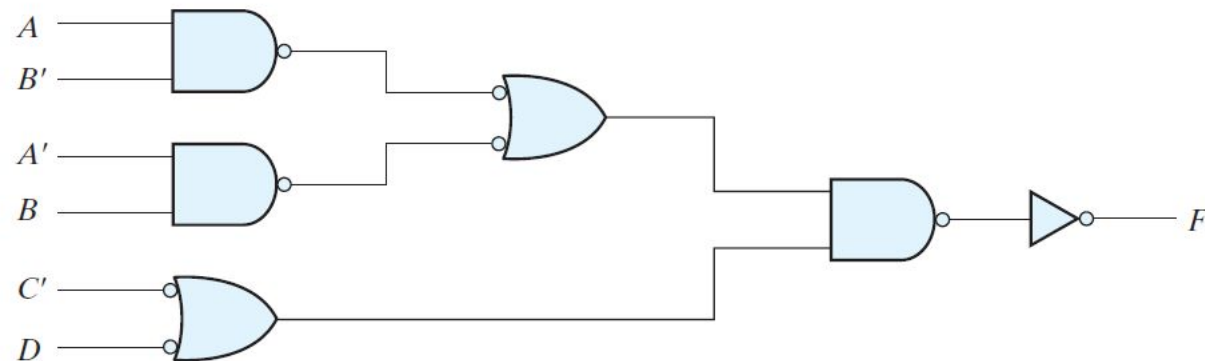


# OTRO EJEMPLO DE IMPLEMENTACIÓN DE UNA FUNCIÓN USANDO NAND

$$F = (AB' + A'B)(C + D')$$



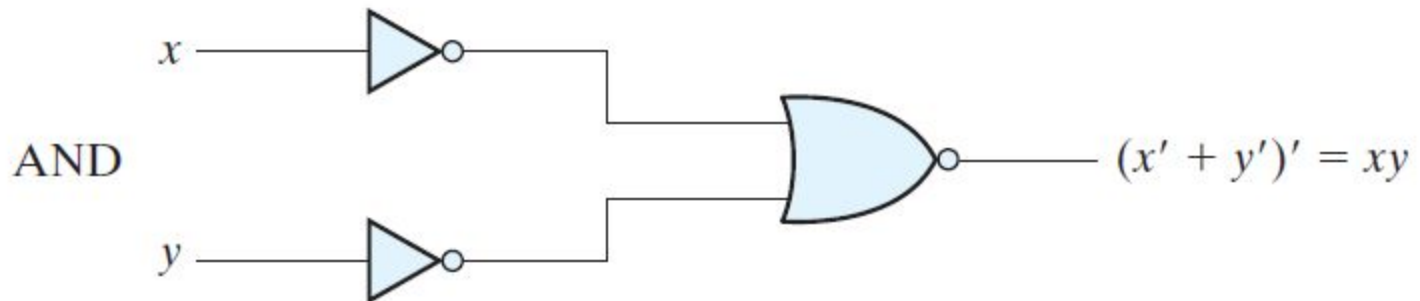
(a) AND-OR gates



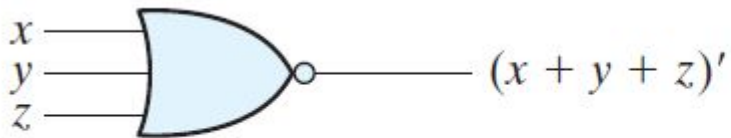
(b) NAND gates

# OPERACIONES LÓGICAS CON COMPUERTAS

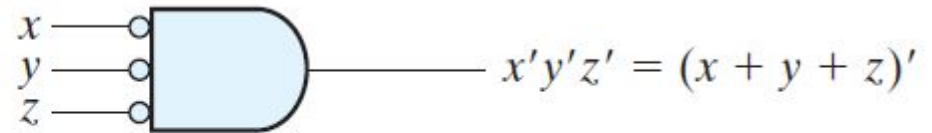
## NOR



# DOS SÍMBOLOS GRÁFICOS PARA REPRESENTAR LA COMPUERTA **NOR**



(a) OR-invert

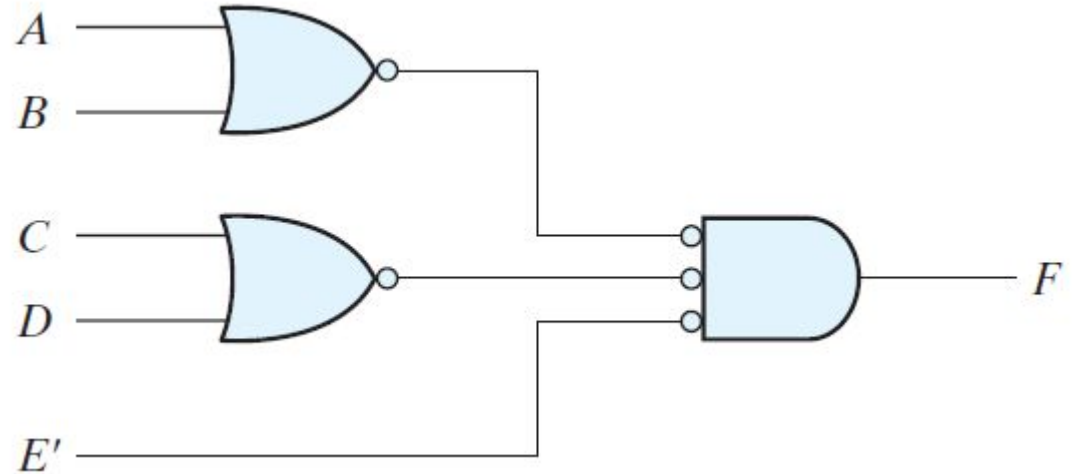


(b) Invert-AND

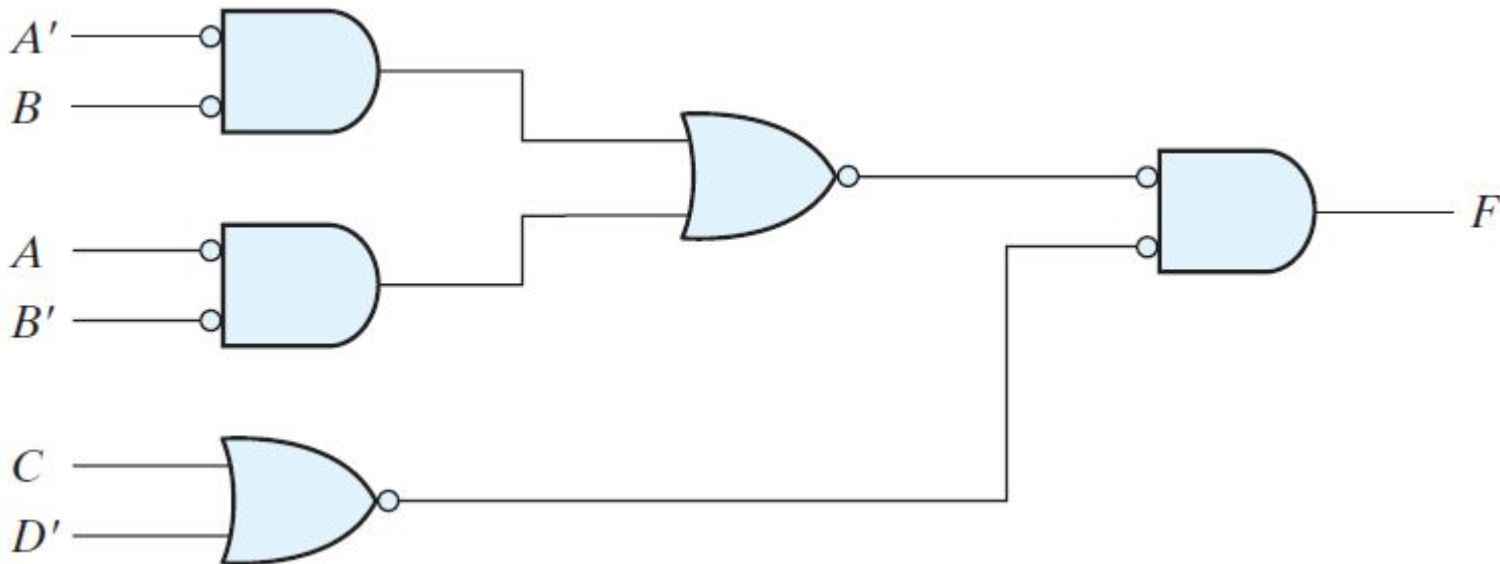


# EJEMPLOS DE IMPLEMENTACIÓN DE FUNCIONES USANDO COMPUERTAS NOR

$$F = (A + B)(C + D)E$$



$$F = (AB' + A'B)(C + D')$$



# FUNCIÓN XOR

| x | y | f |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

$$x \oplus y = xy' + x'y$$

La OR exclusiva es igual a 1 si únicamente **x** es igual a 1 o si únicamente **y** es igual a 1 (es decir, **x** e **y** difieren en su valor).

La NOR exclusiva, también conocida como *equivalencia*, realiza la siguiente operación Booleana:

| x | y | f |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

$$(x \oplus y)' = xy + x'y'$$

La NOR exclusiva es igual a 1 si tanto **x** como **y** son iguales a 1 o si ambas son iguales a 0.

# FUNCIÓN XOR

Las siguientes identidades se aplican a la operación OR exclusiva:

$$x \oplus 0 = x$$

$$x \oplus 1 = x'$$

$$x \oplus x = 0$$

$$x \oplus x' = 1$$

$$x \oplus y' = x' \oplus y = (x \oplus y)'$$

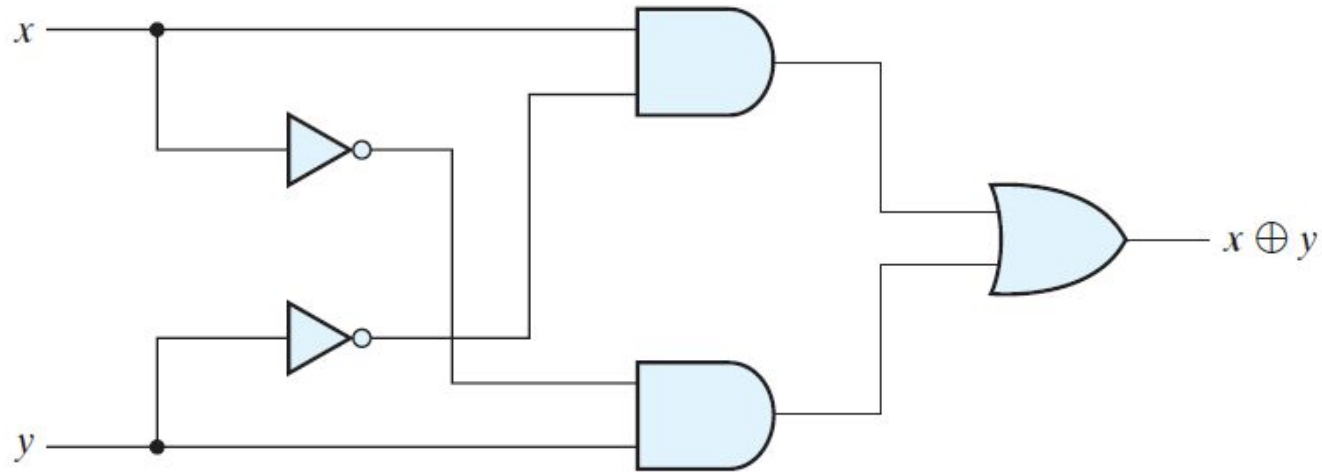
| x | y | f |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

La OR exclusiva es conmutativa y asociativa.

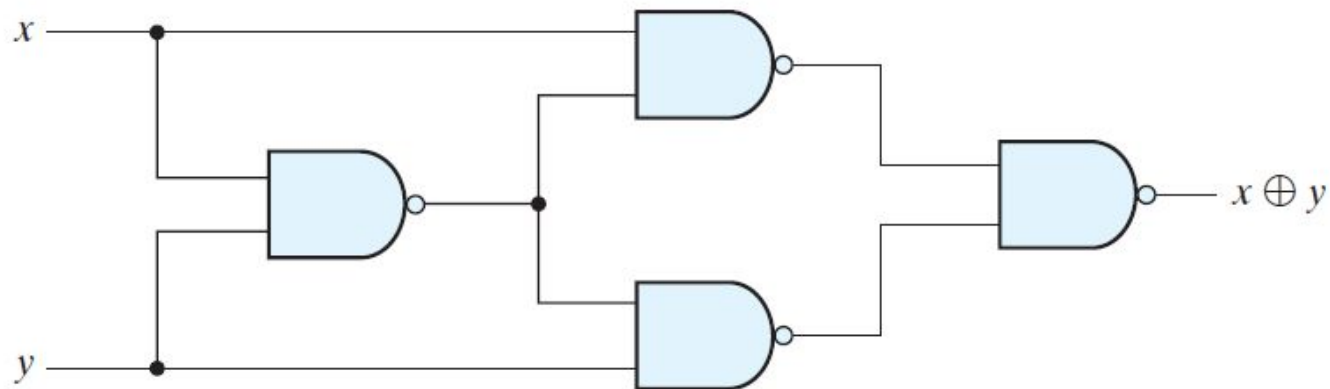
$$A \oplus B = B \oplus A$$

$$(A \oplus B) \oplus C = A \oplus (B \oplus C) = A \oplus B \oplus C$$

# IMPLEMENTACIÓN DE LA FUNCIÓN **XOR**



(a) Exclusive-OR with AND-OR-NOT gates



(b) Exclusive-OR with NAND gates

# FUNCIÓN IMPAR

La operación OR exclusiva con tres o mas variables puede convertirse en un función Booleana común reemplazando el símbolo  $\oplus$  con su equivalente expresión Booleana.

$$\begin{aligned} A \oplus B \oplus C &= (AB' + A'B)C' + (AB + A'B')C \\ &= AB'C' + A'BC' + ABC + A'B'C \\ &= \Sigma(1, 2, 4, 7) \end{aligned}$$

En el caso de tres o mas variables el requerimiento es que un número impar de variables sea igual a 1.

En general, una OR exclusiva de n variables es una función impar definida como la suma lógica de los  $2^n/2$  minitérminos cuyo valor numérico binario tenga un número impar de 1s.

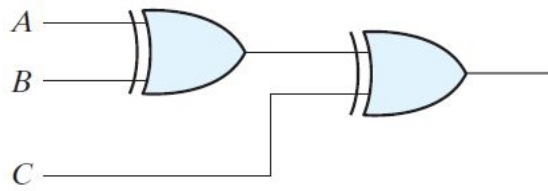
# LA FUNCIÓN **XOR** Y EL MAPA DE KARNAUGH

| $A \backslash BC$ |   | $B$   |       |       |       |
|-------------------|---|-------|-------|-------|-------|
|                   |   | 00    | 01    | 11    | 10    |
| $A$               | 0 | $m_0$ | $m_1$ | $m_3$ | $m_2$ |
|                   | 1 | $m_4$ | $m_5$ | $m_7$ | $m_6$ |
|                   |   | 1     |       | 1     |       |

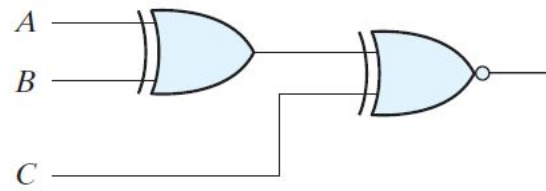
(a) Odd function  $F = A \oplus B \oplus C$

| $A \backslash BC$ |   | $B$   |       |       |       |
|-------------------|---|-------|-------|-------|-------|
|                   |   | 00    | 01    | 11    | 10    |
| $A$               | 0 | $m_0$ | $m_1$ | $m_3$ | $m_2$ |
|                   | 1 | $m_4$ | $m_5$ | $m_7$ | $m_6$ |
|                   |   | 1     |       |       | 1     |

(b) Even function  $F = (A \oplus B \oplus C)'$



(a) 3-input odd function



(b) 3-input even function

# LA FUNCIÓN **XOR** Y EL MAPA DE KARNAUGH

| AB \ CD |    | C          |               |               |               |
|---------|----|------------|---------------|---------------|---------------|
|         |    | 00         | 01            | 11            | 10            |
| A       | 00 | $m_0$      | $m_1$<br>1    | $m_3$         | $m_2$<br>1    |
|         | 01 | $m_4$<br>1 | $m_5$         | $m_7$<br>1    | $m_6$         |
|         | 11 | $m_{12}$   | $m_{13}$<br>1 | $m_{15}$      | $m_{14}$<br>1 |
|         | 10 | $m_8$<br>1 | $m_9$         | $m_{11}$<br>1 | $m_{10}$      |
|         |    | D          |               |               |               |

(a) Odd function  $F = A \oplus B \oplus C \oplus D$

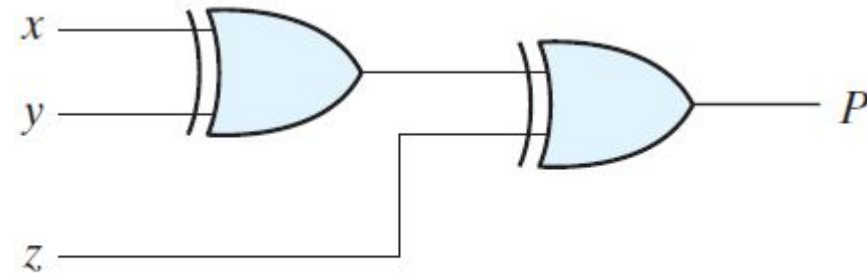
| AB \ CD |    | C             |            |               |               |
|---------|----|---------------|------------|---------------|---------------|
|         |    | 00            | 01         | 11            | 10            |
| A       | 00 | $m_0$<br>1    | $m_1$      | $m_3$<br>1    | $m_2$         |
|         | 01 | $m_4$         | $m_5$<br>1 | $m_7$         | $m_6$<br>1    |
|         | 11 | $m_{12}$<br>1 | $m_{13}$   | $m_{15}$<br>1 | $m_{14}$      |
|         | 10 | $m_8$         | $m_9$<br>1 | $m_{11}$      | $m_{10}$<br>1 |
|         |    | D             |            |               |               |

(b) Even function  $F = (A \oplus B \oplus C \oplus D)'$

# GENERACIÓN Y COMPROBACIÓN DE PARIDAD

*Even-Parity-Generator Truth Table*

| Three-Bit Message |     |     | Parity Bit |
|-------------------|-----|-----|------------|
| $x$               | $y$ | $z$ | $P$        |
| 0                 | 0   | 0   | 0          |
| 0                 | 0   | 1   | 1          |
| 0                 | 1   | 0   | 1          |
| 0                 | 1   | 1   | 0          |
| 1                 | 0   | 0   | 1          |
| 1                 | 0   | 1   | 0          |
| 1                 | 1   | 0   | 0          |
| 1                 | 1   | 1   | 1          |



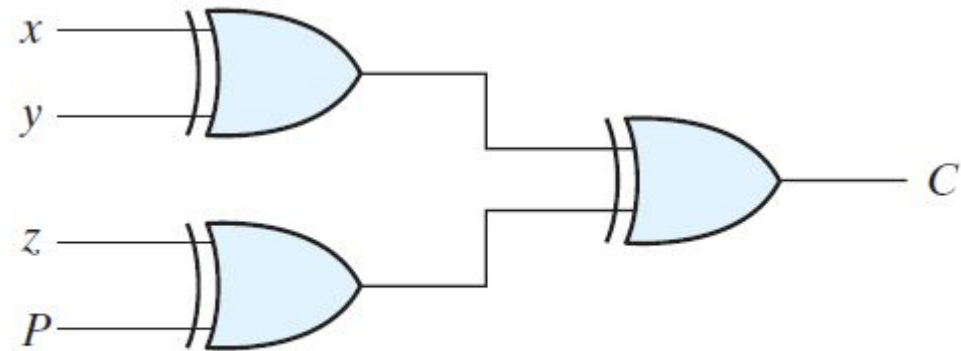
(a) 3-bit even parity generator



# GENERACIÓN Y COMPROBACIÓN DE PARIDAD

*Even-Parity-Checker Truth Table*

| Four Bits Received |     |     |     | Parity Error Check |
|--------------------|-----|-----|-----|--------------------|
| $x$                | $y$ | $z$ | $P$ | $C$                |
| 0                  | 0   | 0   | 0   | 0                  |
| 0                  | 0   | 0   | 1   | 1                  |
| 0                  | 0   | 1   | 0   | 1                  |
| 0                  | 0   | 1   | 1   | 0                  |
| 0                  | 1   | 0   | 0   | 1                  |
| 0                  | 1   | 0   | 1   | 0                  |
| 0                  | 1   | 1   | 0   | 0                  |
| 0                  | 1   | 1   | 1   | 1                  |
| 1                  | 0   | 0   | 0   | 1                  |
| 1                  | 0   | 0   | 1   | 0                  |
| 1                  | 0   | 1   | 0   | 0                  |
| 1                  | 0   | 1   | 1   | 1                  |
| 1                  | 1   | 0   | 0   | 0                  |
| 1                  | 1   | 0   | 1   | 1                  |
| 1                  | 1   | 1   | 0   | 1                  |
| 1                  | 1   | 1   | 1   | 0                  |



(b) 4-bit even parity checker