TECNICAS Y ESTRUCTURAS DIGITALES

PROFESORES MG. SERGIO L. MARTINEZ - ING. VICTOR D. SANCHEZ R.

2022

Unidad 2.2 – ÁLGEBRA DE LOS CIRCUITOS DIGITALES

FUNCIONES LÓGICAS Y GRUPOS LÓGICOS

Contenido

- Funciones lógicas
  - Definición
  - Tablas de verdad
  - Diferentes formatos
- Funciones básicas de 1 variable
- Funciones básicas de 2 variables
  - Ejemplos
- Grupos lógicos completos
  - Chips con compuertas
- Compuertas universales NOR y NAND

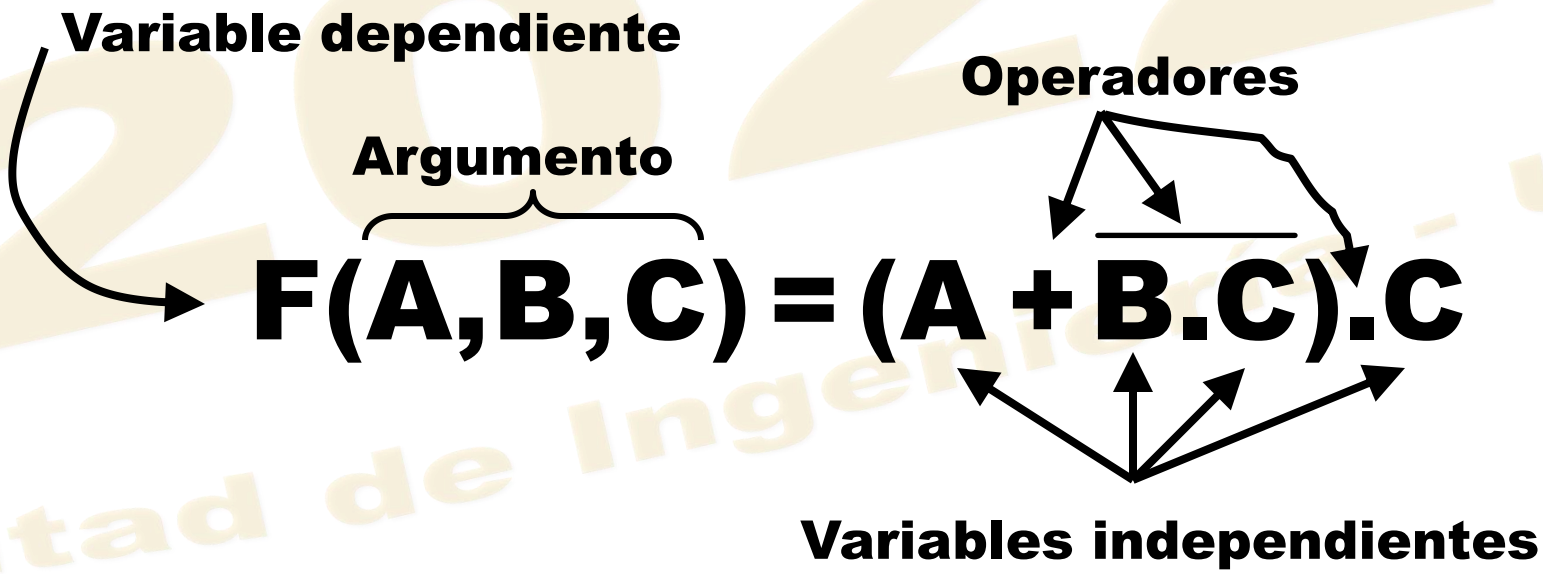
Referencias (Aula Virtual)

- Flórez Fernández H. (2010). DISEÑO LÓGICO. Capítulo 2: Compuertas Lógicas; Capítulo 3: Álgebra de Boole.
- Mano M., Kime C., (2005). FUNDAMENTOS DE DISEÑO LÓGICO Y DE COMPUTADORAS. Cap. 2: Circuitos Lógicos Combinacionales.
- Mandado E., (1998). SISTEMAS ELÉCTRÓNICOS DIGITALES. Capítulo 2: Álgebra de Boole; Capítulo 3: Sistemas Combinacionales.
- Angulo Usategui (1991). ELECTRÓNICA DIGITAL MODERNA - Teoría y Práctica. Cap. 3: El Álgebra Lógica o de Boole.

Definición

(Mandado, pg. 29)

Es la relación que se establece entre una o más variables lógicas (identificadas como independientes o de entrada), mediante los operadores lógicos, cuyo resultado se asigna a una nueva variable lógica (identificada como dependiente o de salida).



Importante: En este campo algebraico, los signos u operadores [+ ; . ; *] NO representan suma y producto. Así también, las variables lógicas (A, B, C...) no contienen números (o bits), sinó estados de eventos.

Tabla de verdad

(Mandado, pg. 33)

Es una organización tabular que contiene todas las posibles combinaciones de valores que pueden establecerse entre las variables de entrada y los correspondientes valores que asume la o las variables de salida.

Entradas			Salida
A	B	C	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

$2^{N°\ variables}$

A=1 , B=0 , C=1

$(A+\overline{B.C}).C = (1+\overline{0.1}).1 = 1$

Tabla de verdad – Otros formatos

T.V. extendida: Contiene los resultados parciales de la función para facilitar el cálculo.

A	B	$\bar{B}$	A+B	$A+\bar{B}$	$(A+B)+(A+\bar{B})$
0	0	1	0	1	0
0	1	0	1	0	0
1	0	1	1	1	1
1	1	0	1	1	1

T.V. horizontal: Ocupa menos espacio cuando hay muchas variables de entrada.

X	0	0	0	0	1	1	1	1
Y	0	0	1	1	0	0	1	1
Z	0	1	0	1	0	1	0	1
F	1	1	0	0	0	0	1	1

T.V. no gráfica: Se representa en forma numérica sin una estructura tabular gráfica. Es útil en programación.

(XYZ,F) = (000,1) (001,1) (010,0) (011,0) (100,0) (101,0) (110,1) (111,1)

En general, cualquier configuración que presente a *n* variables y la referencia a 2^n combinaciones binarias (todas), asociadas a los valores de la/las variables de salida, puede considerarse una tabla de verdad.

Funciones básicas de una variable

- **Representan la asociación entre los dos bits que puede tomar una variable lógica de entrada y todas las posibles combinaciones que puede asumir una variable lógica de salida, para las entradas indicadas, que serán reconocidas como funciones lógicas básicas.**

Variable de entrada				
A	0	1		
Variable de salida			Función	Descripción
f_1	0	0	$f_1 = 0$	función falsedad.
f_2	0	1	$f_2 = A$	función YES.
f_3	1	0	$f_3 = \overline{A}$	función NOT.
f_4	1	1	$f_4 = 1$	función verdad.

$$F = f_i(A)$$

Se interpreta como:

Si A = 0 luego $f_3 = 1$ y si A = 1 luego $f_3 = 0$ entonces f_3 es función NOT (negación)

Funciones básicas de dos variables

En este caso se tienen 2 variables de entrada, las que producen 4 pares de valores binarios. Luego, cada par binario de entrada puede asociarse a un '1' o a un '0' contenido en la variable de salida.

Generando todas las posibles combinaciones para los 4 pares binarios, se generan 16 líneas de resultados, como muestra la tabla.

La interpretación de cada tipo de función, es similar al caso anterior.

— **no aplicables en el álgebra binaria por no cumplir con la propiedad conmutativa.**

$F = f_i(A,B)$

Variables de entrada						
A	0	0	1	1		
B	0	1	0	1		
Variable de salida					Función	Descripción
f_1	0	0	0	0	$f_1 = 0$	función falsedad.
f_2	0	0	0	1	$f_2 = A.B$	función AND.
~~f_3~~	~~0~~	~~0~~	~~1~~	~~0~~	~~$f_3 = \overline{A \Rightarrow B}$~~	~~función A implica B negada.~~
f_4	0	0	1	1	$f_4 = A$	
~~f_5~~	~~0~~	~~1~~	~~0~~	~~0~~	~~$f_5 = \overline{B \Rightarrow A}$~~	~~función B implica A negada.~~
f_6	0	1	0	1	$f_6 = B$	
f_7	0	1	1	0	$f_7 = A \oplus B$	función O-exclusiva (XOR).
f_8	0	1	1	1	$f_8 = A + B$	función OR.
f_9	1	0	0	0	$f_9 = \overline{A+B}$	función NOR.
f_{10}	1	0	0	1	$f_{10} = \overline{A \oplus B}$	función coincidencia (XNOR).
f_{11}	1	0	1	0	$f_{11} = \overline{B}$	
~~f_{12}~~	~~1~~	~~0~~	~~1~~	~~1~~	~~$f_{12} = B \Rightarrow A$~~	~~función B implica A.~~
f_{13}	1	1	0	0	$f_{13} = \overline{A}$	
~~f_{14}~~	~~1~~	~~1~~	~~0~~	~~1~~	~~$f_4 = A \Rightarrow B$~~	~~función A implica B.~~
f_{15}	1	1	1	0	$f_{15} = \overline{A.B}$	función NAND.
f_{16}	1	1	1	1	$f_{16} = 1$	función verdad.

Ejemplo de función básica

(Mandado, pg. 34; Mano, pg. 65)

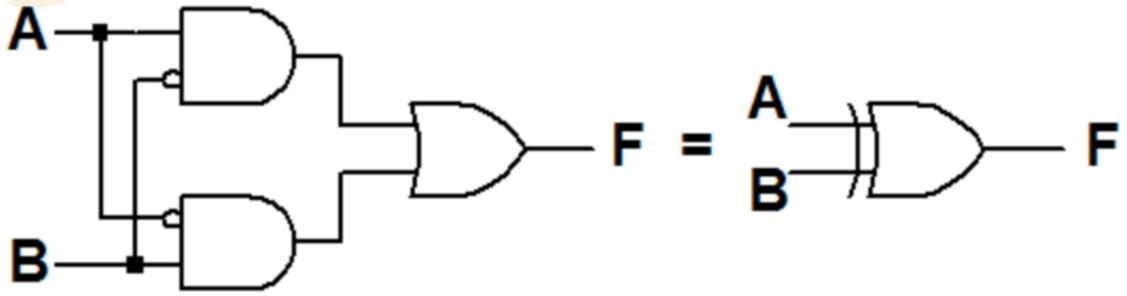
Función O-Exclusiva

- **También llamada XOR, se define para dos variables como la función que asume una condición verdadera (1) cuando las variables presentan distintos estados.**
- **Definida para más de dos variables es la función que asume una condición verdadera (1) cuando hay un número impar de variables en estado (1).**
- **Su utilización más difundida es para construir sumadores numéricos.**

Forma normal $F(A,B) = A \oplus B$

Forma desarrollada $F(A,B) = A.\overline{B} + \overline{A}.B$

A	B	F
0	0	0
0	1	1
1	0	1
1	1	0



Ejemplo de función básica

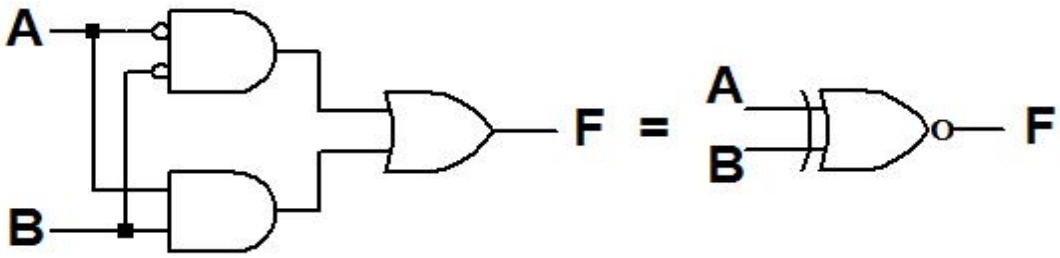
Función NOR-Exclusiva

- Conocida también como coincidencia, equivalencia o XNOR, resulta de la negación de la función XOR.
- Se define para dos variables como la función que asume un estado verdadero (1) cuando las variables presentan los mismos estados.
- Para más de dos variables, es la función que asume un estado verdadero (1) cuando hay un número par de variables que se encuentren en este estado.
- Su uso más difundido es para construir comparadores.

Forma normal $F(A,B) = \overline{A \oplus B}$

Forma desarrollada $F(A,B) = A \cdot B + \overline{A} \cdot \overline{B}$

A	B	F
0	0	1
0	1	0
1	0	0
1	1	1



Definición y concepto

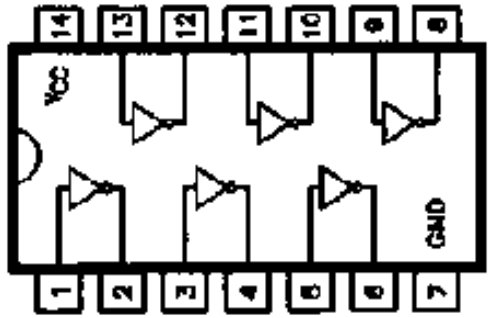
(Mandado, pg. 65; Angulo, pg. 73)

Un grupo lógico completo es el operador o conjunto de operadores lógicos con los que se puede representar cualquier función lógica.

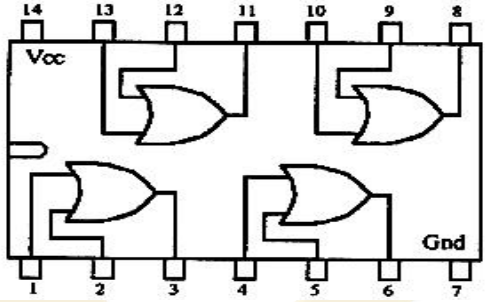
- **Básicamente se definen tres grupos lógicos completos**
  1. **NOT – OR – AND (operadores primarios)**
  2. **NOR (operador secundario o universal)**
  3. **NAND (operador secundario o universal)**
- **Las compuertas basadas en los operadores NOR y NAND también se denominan *compuertas universales*.**
- **Cualquier función lógica tiene formas equivalentes utilizando los grupos 2 y 3 exclusivamente.**
- **La utilidad principal es la construcción de circuitos lógicos que emplean un mismo tipo de compuerta.**
- **Las funciones y circuitos que utilizan los grupos 2 y 3 son más homogéneas pero suelen ser más extensas.**

Compuertas en circuitos integrados (pinout)

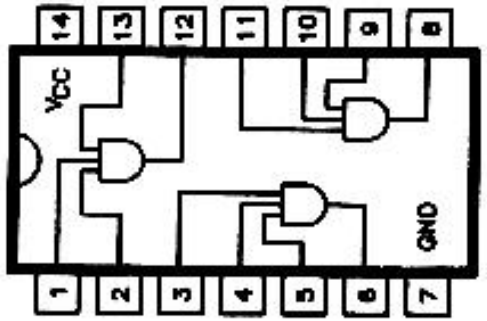
Grupo lógico 1



séxtuple inversor



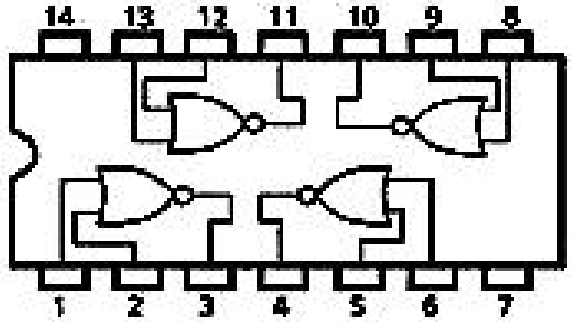
cuádruple 2-input OR

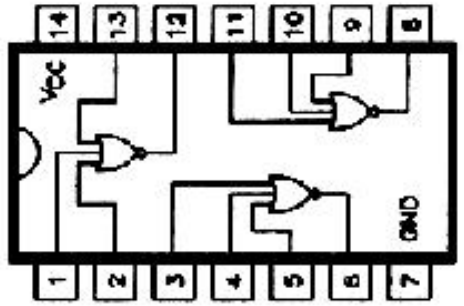


riple 3-input AND

Grupo lógico 2

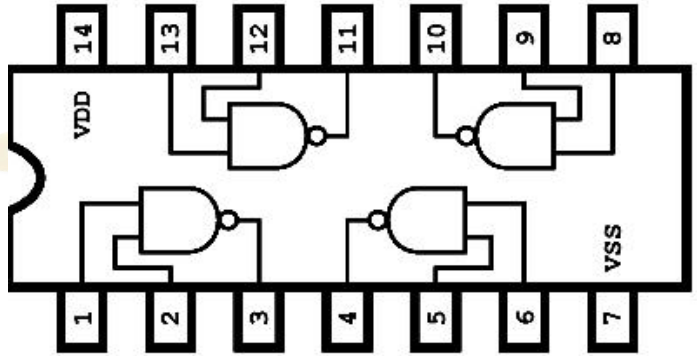
cuádruple 2-input NOR



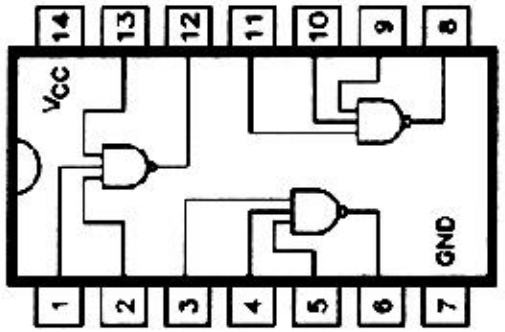


triple 3-input NOR

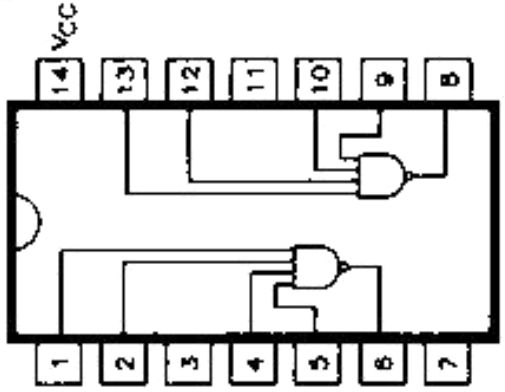
Grupo lógico 3



cuádruple 2-input NAND



triple 3-input NAND



doble 4-input NAND

GRUPOS LÓGICOS COMPLETOS

Criterios para representación con NOR / NAND

- Siempre que sea posible se debe trabajar con la función minimizada.
- Desarrollar los operadores combinados (XOR / XNOR) a sus representaciones NOT / OR / AND.

$$\left.\begin{array}{l} A \oplus B = A.\bar{B} + \bar{A}.B \\ \overline{A \oplus B} = \bar{A}.\bar{B} + A.B \end{array}\right\}$$

- La utilidad principal es la construcción de circuitos lógicos que emplean un mismo tipo de compuerta.
- Aplicar las propiedades en forma conveniente (a toda o parte de la función) para eliminar los operadores no deseados.
- Para formatos con NOR debe buscarse la eliminación de operadores AND y NAND (2° ley de De Morgan).
- Para formatos con NAND debe buscarse la eliminación de operadores OR y NOR (1° ley de De Morgan).

Ejemplo de conversión NOR / NAND

$$F(A,B,C) = (A + \overline{B.C}) \,.\, C$$

Con NOR

- De Morgan

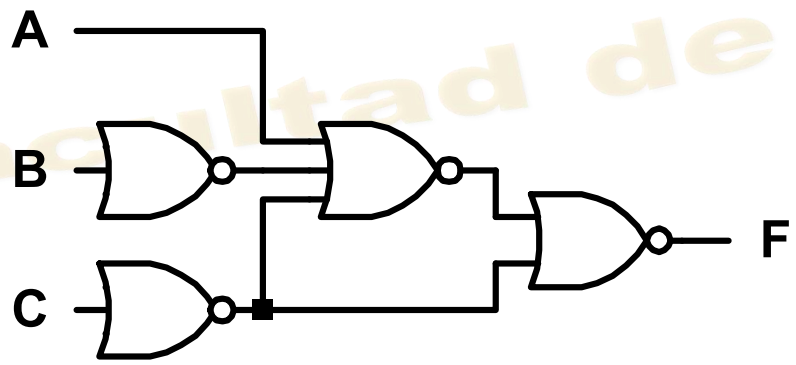
$$F(A,B,C) = (A + \bar{B} + \bar{C}) \,.\, C$$

- Involución

$$F(A,B,C) = \overline{\overline{(A + \bar{B} + \bar{C}) \,.\, C}}$$

- De Morgan al producto lógico

$$F(A,B,C) = \overline{\overline{A + \bar{B} + \bar{C} + \bar{C}}}$$



Con NAND

- Involución al paréntesis

$$F(A,B,C) = \overline{\overline{(A + \overline{B.C}) \,.\, C}}$$

- De Morgan al paréntesis

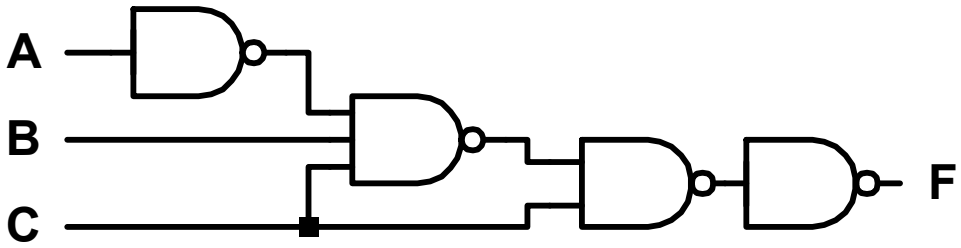
$$F(A,B,C) = \overline{\bar{A} \,.\, \overline{\overline{B.C}} \,.\, C}$$

- Involución inversa

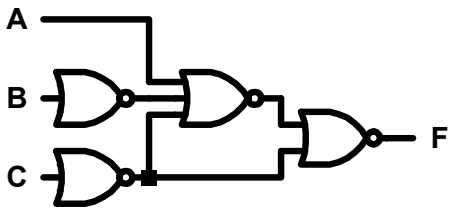
$$F(A,B,C) = \overline{\bar{A} \,.\, B \,.\, C \,.\, C}$$

- Involución a toda la función

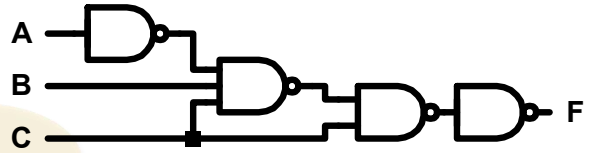
$$F(A,B,C) = \overline{\overline{\bar{A} \,.\, B \,.\, C \,.\, C}}$$



Ejemplo de conversión NOR / NAND con C.I.

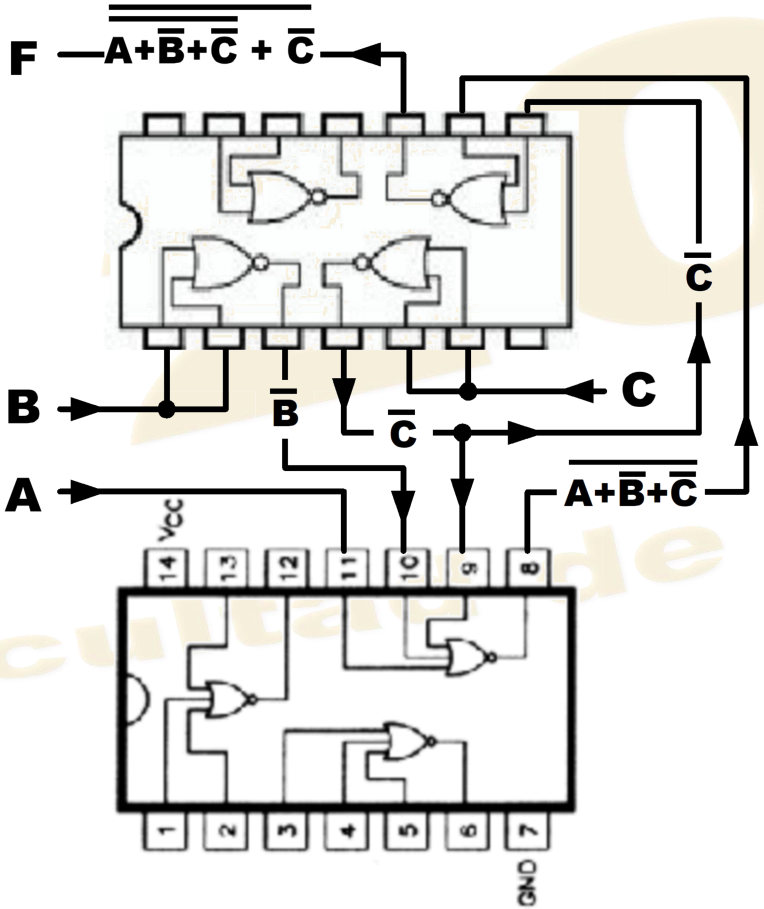


$$F(A,B,C) = (A + \overline{B.C}) \cdot C$$



Con NOR

$$F(A,B,C) = \overline{\overline{A + \overline{B} + \overline{C} + \overline{C}}}$$



Con NAND

$$F(A,B,C) = \overline{\overline{A} \cdot B \cdot C \cdot C}$$