

UNIVERSIDAD TÉCNICA ESTATAL DE QUEVEDO



FACULTAD DE CIENCIAS DE LA COMPUTACIÓN Y DISEÑO DIGITAL

TEMA

EJERCICIOS SOBRE SIMPLIFICACIÓN DE EXPRESIONES

INTEGRANTES

LLERENA ABRIL ANGELINA JULEXY

MORA DUARTE ALEX JOSE

MORALES SANCHEZ GARY ALEJANDRO

ZAMORA AGUILAR RONALDO WILFRIDO

CURSO

2DO SOFTWARE “B”

GRUPO

C

MATERIA

ARQUITECTURA DE COMPUTADORAS

Índice

1. INVESTIGACIONES BIBLIOGRÁFICAS	3
1.1 Aritmética binaria	3
1.2 Aritmética decimal	3
1.2.1 Ejemplo de aritmética decimal	3
1.3 Álgebra de circuitos digitales	4
1.3.1 Álgebra booleana.....	4
1.3.2 Simplificación de expresiones booleanas	5
1.3.3 Leyes del álgebra booleana.....	5
1.3.4 Formas Canónicas del Álgebra Booleana (SOP y POS)	6
1.4 Función de Boole.....	7
1.5 Circuitos lógicos	8
1.6 Circuitos lógicos de sistemas digitales	11
2. EJERCICIOS PROPUESTOS POR EL PROFESOR	13
2.1 Ejercicio 3-16	13
2.2 Ejercicio 3-17	15
2.3 Ejercicio 3-26	16
2.4 Ejercicio 3-38	17
2.5 Ejercicio 4-11	18
2.6 Ejercicio 4-12	19
3. BIBLIOGRAFÍA	20
4. ANEXO	21

1. INVESTIGACIONES BIBLIOGRÁFICAS

1.1 Aritmética binaria

Para Roero y Luciano (2021), la aritmética binaria representa un punto de encuentro con las matemáticas, filosofía y tecnología, dado que Gottfried Wilhelm Leibniz formuló el sistema binario como una forma lógica y universal de representar los números. Además, habla cómo Giuseppe Peano, siglos después, toma estas ideas y las aplica a la taquigrafía, construyendo un sistema que une lógica binaria, lenguaje y técnica mecánica [1].

En cambio, Natarajan et al. (2020) habla del papel funcional de la aritmética binaria en la electrónica digital. Describe cómo operaciones como suma o resta, multiplicación y división se ejecutan mediante circuitos como los sumadores binarios, y destaca el uso del complemento a dos para representar números negativos [2].

Chang et al. (2021) señalan que, incluso con los avances que se ha tenido de la computación cuántica, los principios binarios siguen siendo importantes y fundamentales. A través de puertas cuánticas que reproducen funciones binarias, se demuestra que la lógica binaria no solo tiene un origen histórico y un papel técnico, conserva una presencia activa en las tecnologías más innovadoras y cambios de la actualidad [3].

1.2 Aritmética decimal

Para Mudawar Muhamed (2023), los números decimales son importantes por su aplicación en diversas áreas, como la científica e ingeniería, a diferencia de Tian, Braithwaite y Siegler (2021) que nos dice que la aritmética decimal ha recibido menos importancia de parte de los investigadores [4][5]. Mudawar Muhamed sostiene que los números decimales también son usados habitualmente en cálculos financieros, banca, conversión de divisas e impuestos [4].

1.2.1 Ejemplo de aritmética decimal

Tian, Braithwaite y Siegler (2021) dicen que, para sumar dos números decimales, se debe realizar un método donde implica alinear los dígitos de los sumandos según sus valores posicionales (unidades, decenas, centenas), luego sumar las cifras decimales y después las partes enteras [5].

$$\begin{array}{r} +1 \\ 5,62 \\ + 0,7 \\ \hline 6,32 \end{array}$$

1.3 Álgebra de circuitos digitales

En el libro “Boolean Algebra and Logic Gates” (2021) afirma que, el álgebra de circuitos digitales tiene como base el uso del álgebra booleana como método matemático para poder modelar y diseñar el comportamiento de los sistemas lógicos. A través de esta herramienta, es posible expresar funciones lógicas mediante variables binarias y operadores lógicos básicos, describiendo cómo responden los circuitos digitales ante distintas combinaciones de entrada. Esta representación algebraica resulta esencial para el diseño de compuertas lógicas y estructuras combinacionales en el que la práctica del álgebra booleana ha convertido en el asidero del diseño lógico contemporáneo en la electrónica digital[6].

1.3.1 Álgebra booleana

El libro “Electronics for Physicists” (2020) expone el álgebra booleana como una de las herramientas matemáticas más importantes para analizar los circuitos digitales. El álgebra booleana se fundamenta en los valores binarios, es decir, verdadero (1) o falso (0), que representan los dos estados de la corriente eléctrica: encendido o apagado. A partir de este punto se empieza a describir cómo se comportan las señales digitales que circulan en los diferentes sistemas electrónicos. El libro menciona que el álgebra booleana permite representar y manipular funciones lógicas que determinan el funcionamiento de las compuertas lógicas digitales. El vínculo entre las matemáticas y la electrónica garantiza una comprensión y facilita el diseño de circuitos. Esta álgebra es un lenguaje fundamental a la hora de tratar con lógica digital. Por otra parte, el propio autor presenta los principios elementales de este tipo de álgebra antes de mostrar ejemplos concretos. Por este motivo, establece una fundamentación clara para comprender cómo los circuitos digitales procesan información [7].

1.3.2 Simplificación de expresiones booleanas

El libro “Simplification of logical functions with application to circuits” (2022) nos dice que, la simplificación de expresiones booleanas es uno de los procedimientos necesarios en el diseño eficiente de circuitos digitales, ya que permite reducir el número de compuertas lógicas utilizadas, y en consecuencia optimizar el coste o la velocidad de funcionamiento. Tradicionalmente se ha recurrido a los mapas de Karnaugh o al algoritmo de Quine-McCluskey como métodos de simplificación de funciones lógicas, aunque tienen fuertes limitaciones para manejar grandes cantidades de variables. El artículo de Feng, Zhao y Cui plantea un nuevo procedimiento de simplificación de expresiones booleanas que se basa en el producto semi-tensorial de matrices, para transformar funciones booleanas en formas algebraicas, y para aplicar criterios estructurales que permiten la simplificación. Con el procedimiento introducido la minimización puede automatizarse de forma más sencilla mediante algoritmos. Se superan las limitaciones impuestas por los métodos clásicos en cuanto al número de variables que imponen procedimientos clásicos, por lo que su propuesta puede ser vista como una herramienta altamente eficaz en la simplificación lógica en situaciones complejas. Por último, se demuestra que su método tiene una complejidad computacional aceptable ($O(2^n)$) y se puede incluir dentro de los procesos de diseño de circuitos sin tener que recurrir a representaciones gráficas o tabulares. El algoritmo además tiene en cuenta el orden de las variables lo cual, influye en el nivel de simplificación alcanzado[8].

1.3.3 Leyes del álgebra booleana

Dicho por Erciyes (2021), el álgebra de Boole es un sistema matemático que se basa en variables binarias (0 y 1), fundamentales para el diseño y análisis de circuitos digitales. Dentro de este sistema, se establecen diversas leyes que permiten simplificar expresiones lógicas [9]:

Ley	Expresión	Descripción
Identidad	$a + 0 = a$ $a * 1 = a$	Si se operar con 0 (en suma) o 1 (en producto) no se altera el valor de la variable.
Idempotente	$a + a = a$ $a * a = a$	Si se repite una variable no cambia el resultado.

Complemento	$a + a' = 1$ $a * a' = 0$	Una variable y su negación siempre producen un resultado constante.
Conmutativa	$a + b = b + a$ $a * b = b * a$	El orden de los operandos no afecta el resultado.
Asociativa	$a + (b + c) = (a + b) + c$ $a * (b * c) = (a * b) * c$	El modo en que se agrupan los términos no cambia el valor.
Distributiva	$a * (b + c) = ab + ac$ $a + (b * c) = (a + b) (a + c)$	La multiplicación se distribuye sobre suma (y viceversa).
Absorción	$a + ab = a$ $a (a + b) = a$	Cuando “a” ya está presente, cualquier combinación con “b” no altera el resultado.
De Morgan	$(a + b)' = a' * b'$ $(a * b)' = a' + b'$	Permiten reescribir negaciones de expresiones compuestas.

Siguiendo por lo dicho por Erciyas (2021), estas leyes se aplican bajo el principio de dualidad, el cual indica que cualquier expresión válida en álgebra de Boole sigue siendo válida si se intercambian los operadores $+$ y $*$, así como los valores 0 y 1 [9].

1.3.4 Formas Canónicas del Álgebra Booleana (SOP y POS)

Según Nugroho (2021), dentro del álgebra booleana, se utilizan ampliamente, como expresiones estándares, las conocidas como sum-of-Product (SOP) y Product-of-sum (POS) en la simplificación lógica de funciones. La forma SOP es una suma (OR) de productos llamados minterms, y la forma POS un producto (AND) de sumas (OR) llamados maxterms. Ambas formas requieren que contengan todas las variables del sistema en su forma directa o complementada. El artículo de Nugroho explica que, por ejemplo, una función en forma SOP podría expresarse como $f(x, y, z) = xyz + xyz + xyz$, mientras que en forma POS se representaría como $f(x, y, z) = (x + y + z)(x + y + z)(x + y + z)$. Estas representaciones no solo son esenciales para el análisis teórico,

sino que también constituyen la base sobre la cual operan diversos métodos de simplificación automatizada de funciones booleanas [10].

$$\begin{aligned} \text{Out} = & \overline{A}\overline{B}\overline{C}\overline{D} + \overline{A}\overline{B}\overline{C}D + \overline{A}\overline{B}C\overline{D} \\ & + \overline{A}\overline{B}CD + \overline{A}B\overline{C}\overline{D} + \overline{A}B\overline{C}D + \overline{A}BC\overline{D} \\ & + \overline{A}BCD + AB\overline{C}\overline{D} + AB\overline{C}D + ABC\overline{D} \end{aligned}$$

$$f(A,B,C,D) = \sum m(0,1,3,4,5,7,12,13,15)$$

A \ B	CD			
	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

A \ B	CD			
	00	01	11	10
00	1	1	1	0
01	1	1	1	0
11	1	1	1	0
10	0	0	0	0

A \ B	CD			
	00	01	11	10
00	1	1	1	0
01	1	1	1	0
11	1	1	1	0
10	0	0	0	0

$$f(A,B,C,D) = \overline{A}\overline{C} + \overline{A}D + B\overline{C} + BD$$

Figura 1. Ejemplo práctico de simplificación de función booleana

1.4 Función de Boole

La función booleana para Kurgalin y Borzunov (2020), formalizada como $f: \{0,1\}^n \rightarrow \{0,1\}$ es un elemento esencial en matemáticas, lógica y en el diseño de sistemas digitales. Donde cada una de las n variables determina un valor de salida, lo que modela una estructura sencilla pero poderosa, con la cual podemos conformar decisiones simples y darle forma a la base de los circuitos lógicos. [11]

En el libro “On the numeration of Boolean functions with distinguished variables” Josep Freixas nos presenta siete nuevas fórmulas para resolver cada caso específico de las funciones booleanas, lo que demuestra que el problema no es tan sencillo como parece, el cual es conocido como problema de Dedekind. Este tipo de estudios es relevante para entender todas las variedades de diseños en casos donde la función booleana debe cumplir con ciertas propiedades como la de monotonicidad o canalización. [12]

Otro estudio relevante lo encontremos en el libro “Boolean Differential Calculus” donde Bernd Steinbach y Christian Posthoff entran más en profundidad sobre el análisis de las funciones booleanas aplicando métodos similares al cálculo diferencial. Con esto se pudo verificar cómo reaccionan las funciones booleanas con el cambio de sus variables, aplicando incluso derivadas lógicas. Lo cual tuvo un gran impacto en el diseño de circuitos con menos retrasos, reduciendo el consumo energético y para el testeo automático de fallos. [13]

En la estructura algebraica y gráfica de las funciones booleanas, el capítulo “Boolean Functions with a Few Walsh Transform Values” estudia funciones en las que sus transformaciones de Walsh muestran pocos valores distintos. Estas funciones son importantes en comunicaciones y criptografía por su no linealidad y propiedades de correlación bien definidas. [14]

También el libro “Boolean Algebras and Combinational Circuits” introduce los fundamentos algebraicos, derivadas vectoriales y ecuaciones diferenciales booleanas. Estas herramientas permiten cambiar una función incompleta en una forma función que sea evaluable y lista para ser utilizada, esto nos sirve para simplificar los circuitos digitales, optimizar la cobertura y reducir la dificultad de la síntesis lógica. [15]

Finalmente tenemos el libro “Boolean Functions” el cual ofrece una visión más simplificada sobre las funciones booleanas. En este podemos observar cómo estas funciones booleanas no solo sirven para los circuitos de conmutación, sino que también son importantes en otros aspectos como los algoritmos criptográficos y los lenguajes de programación, subrayando su adaptabilidad y su alcance en la actualidad. [16]

1.5 Circuitos lógicos

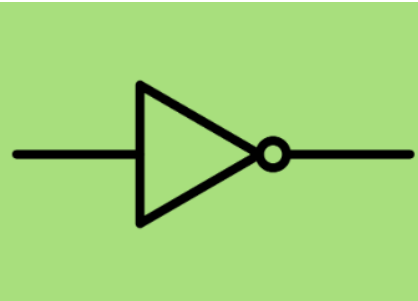
Dicho por Ward Hubert (2024), todos los circuitos lógicos usan entradas que producen salidas. Normalmente, las entradas son representadas por letras, como: A, B, C, etc. Y estas entradas pueden tener solamente 2 posibles salidas: 0 o 1 [17].

Según LaMeres Brock (2024), Un símbolo lógico es una representación gráfica para mostrar cómo se conectan los circuitos del sistema, cada uno de los símbolos que existen tienen una forma única que indican gráficamente su funcionalidad [18]. Ward Hubert afirma que las puertas lógicas tienen funciones como NOT, AND, OR, NAND y NOR [17].

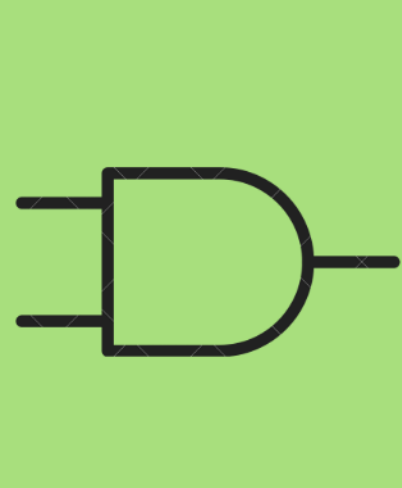
Siguiendo con lo dicho por LaMeres Brock (2024), La tabla de verdad se utiliza para representar el comportamiento de un circuito lógico [18].

A continuación, se mostrarán el comportamiento de las puertas lógicas según LaMeres Brock (2024):

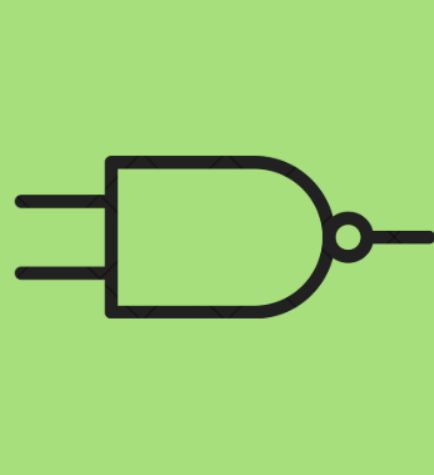
NOT

	A	OUT
	0	1
	1	0


AND

	A	B	OUT
	0	0	0
	0	1	0
	1	0	0
	1	1	1


NAND

	A	B	OUT
	0	0	1
	0	1	1
	1	0	1
	1	1	0


OR

	A	B	OUT
	0	0	0
	0	1	1
	1	0	1
	1	1	1


NOR

	A	B	OUT
	0	0	1
	0	1	0
	1	0	0
	1	1	0

XOR

	A	B	OUT
	0	0	0
	0	1	1
	1	0	1
	1	1	0

XNOR



A	B	OUT
0	0	1
0	1	0
1	0	0
1	1	1

1.6 Circuitos lógicos de sistemas digitales

Según Kaufmann (2020), Los circuitos lógicos son la base fundamental del funcionamiento de los sistemas digitales modernos, ya que permiten implementar operaciones aritméticas y lógicas esenciales en procesadores, memorias y dispositivos electrónicos [19]. Para Kaufmann (2020), el diseño automatizado de estos circuitos, especialmente los de tipo aritmético, puede mejorarse considerablemente mediante el uso de álgebra computacional. Este enfoque no solo permite representar con precisión el comportamiento lógico de las compuertas, sino también verificar, optimizar e incluso reconstruir funciones complejas que forman parte del circuito [19].

Para Sampathkumar et al. (2025), los circuitos lógicos aritméticos pueden optimizarse significativamente mediante el uso de técnicas algebraicas, donde cada compuerta lógica se representa como un polinomio. Este enfoque algebraico permite modelar el comportamiento del circuito de forma precisa, facilitando la identificación y corrección de errores internos a través de la síntesis parcial, especialmente en estructuras complejas como los multiplicadores enteros. Además, el método propuesto permite detectar entradas que no afectan el funcionamiento del circuito (don't-care sets), lo cual es útil para reducir el área física, el retardo de propagación y el consumo energético del circuito digital [19].

Por otro lado, Kaufmann (2020) plantea el uso del álgebra computacional como una herramienta clave para automatizar el diseño y la verificación de circuitos aritméticos, haciendo énfasis en el modelado lógico y la síntesis inversa. A través de esto, se puede verificar si un circuito implementa correctamente una operación matemática y, si es necesario, reconstruir su especificación algebraica. Este enfoque resulta especialmente

valioso en aplicaciones donde se requiere alta precisión y eficiencia, como en procesadores, unidades aritméticas y sistemas criptográficos [20].

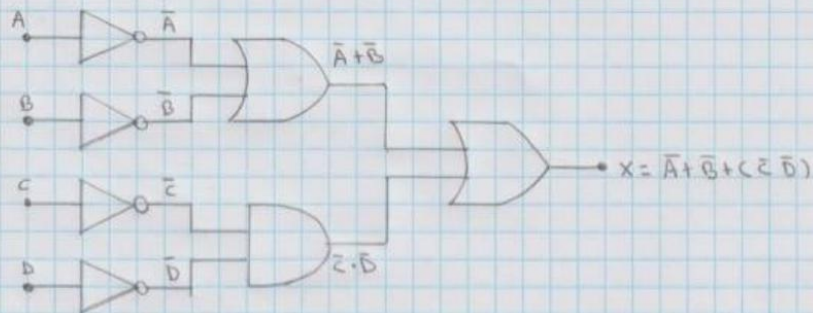
Los estudios demuestran que las herramientas algebraicas mejoran el desarrollo de circuitos lógicos aritméticos. Sampathkumar et al. (2025) corrigen errores internos y optimizan bloques, mientras que Kaufmann (2020) se enfoca en verificación formal y reconstrucción de especificaciones. Estos aportes muestran que el álgebra computacional no solo aumenta la eficiencia y confiabilidad de los diseños digitales, sino que también facilita su corrección y diseño automatizado [19][20].

2. EJERCICIOS PROPUESTOS POR EL PROFESOR

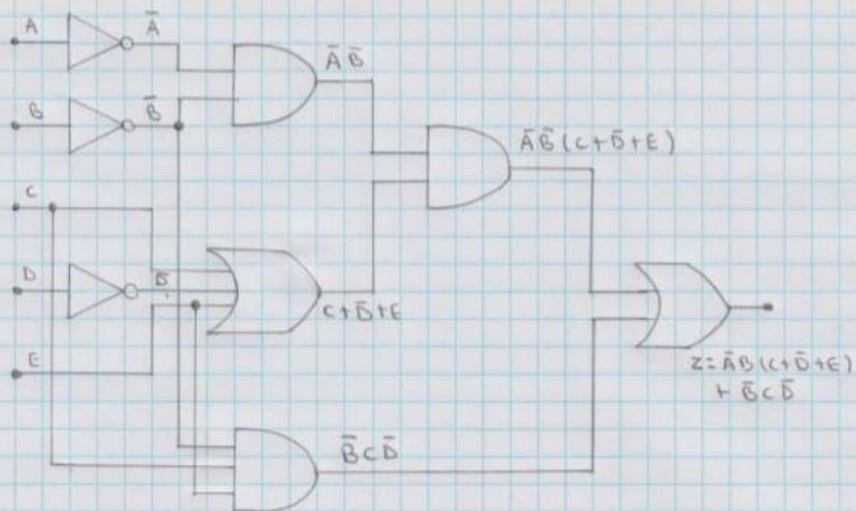
2.1 Ejercicio 3-16

3-16. Para cada una de las siguientes expresiones, construya el circuito lógico correspondiente, utilizando puertas AND y OR e Inversores.

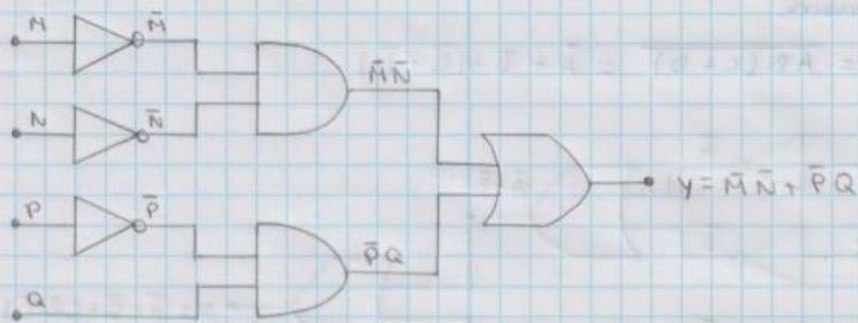
(a) $\star X = \overline{AB(C+D)} = \bar{A} + \bar{B} + (\bar{C} \cdot \bar{D})$



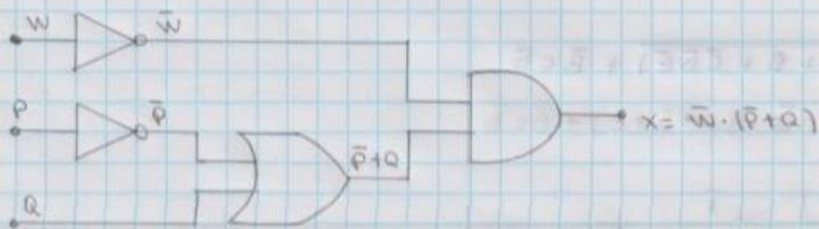
(b) $\star Z = \overline{(A+B+C\bar{D}\bar{E})} + \bar{B}C\bar{D}$
 $= \bar{A} \cdot \bar{B} \cdot (C + \bar{D} + E) + \bar{B}C\bar{D}$



$$(c) y = \overline{(M+N+PQ)} = \bar{M} \cdot \bar{N} + \bar{P} \cdot Q$$



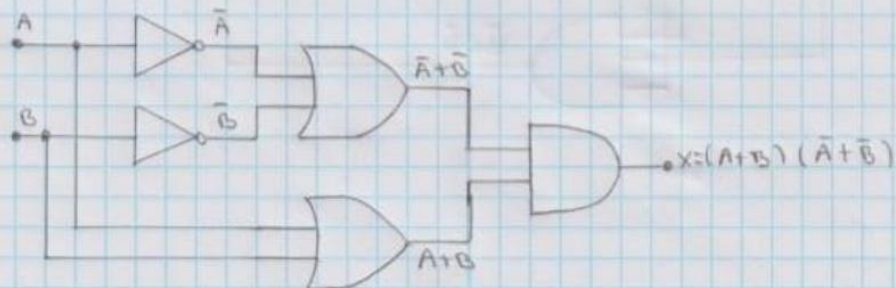
$$(d) x = \overline{W+PQ} = \bar{W} \cdot (\bar{P}+Q)$$



$$(e) z = \overline{MNP(P+\bar{N})} \rightarrow z = MNP + MN\bar{N} \rightarrow MNP + N(0) \rightarrow z = M \cdot N \cdot P$$



$$(f) x = (A+B)(\bar{A}+\bar{B})$$



2.2 Ejercicio 3-17

3-17. Aplique las formas de onda de entrada de la Figura 3-54 a una puerta NOR y dibuje la forma de onda de salida.

① Aplique las formas de onda de entrada de la Figura 3-54 a una puerta NOR y dibuje la forma de onda de salida.

A	B	C	X
1	0	1	0
1	0	0	0
0	1	1	0
0	1	0	0

② Repite con C en posición BAJA

A	B	C	X
1	0	0	1
1	0	0	0
0	1	0	1
0	1	0	0

③ Repite con C en posición ALTA

A	B	C	X
1	0	1	0
1	0	1	0
0	1	1	0
0	1	1	0

2.3 Ejercicio 3-26

3-26 Simplifique cada una de las siguientes expresiones utilizando los teoremas de Morgan.

$$(a) \overline{A B C} = \overline{A} + \overline{B} + \overline{C} = A + \overline{B} + C \quad R_{11}$$

$$(b) \overline{A + B C} = \overline{A} \cdot \overline{(B \cdot C)} = A \cdot (\overline{B} + \overline{C}) \quad R_{11}$$

$$(c) \overline{A B C D} = \overline{A} + \overline{B} + \overline{C} + \overline{D} = \overline{A} + \overline{B} + C + D$$

$$(d) \overline{A + B} = \overline{A} \cdot \overline{B} = \overline{A \cdot B} \quad R_{11}$$

$$(e) \overline{\overline{A B}} = \overline{\overline{A} \cdot \overline{B}} = A \cdot B \quad R_{11}$$

$$(f) \overline{\overline{A} + \overline{C} + \overline{D}} = \overline{\overline{A}} \cdot \overline{\overline{C}} \cdot \overline{\overline{D}} = A \cdot C \cdot D \quad R_{11}$$

$$(g) \overline{A(B + \overline{C})D} = \overline{A \cdot B \cdot \overline{C} \cdot D} = \overline{A \cdot \overline{B} \cdot C \cdot D} \\ = \overline{A} + \overline{\overline{B}} + \overline{C} + \overline{D} = \overline{A} + B + \overline{C} + \overline{D} \quad R_{11}$$

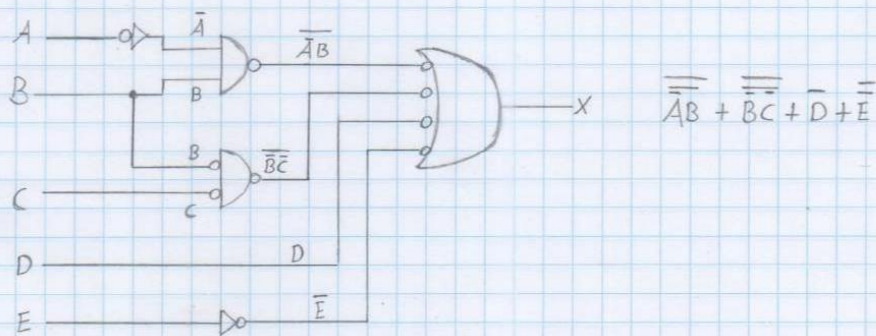
$$(h) \overline{(M + \overline{N})(\overline{M} + N)} = \overline{(M + \overline{N}) + (\overline{M} + N)} = \overline{(M \cdot \overline{N}) + (\overline{M} \cdot N)} \\ = \overline{(M \cdot N)} + \overline{(M \cdot \overline{N})} = \overline{M}N + M\overline{N} \quad R_{11}$$

$$(i) \overline{\overline{A B C D}} = \overline{(\overline{A + B})C} \rightarrow A \cdot B + \overline{C} \\ = \overline{(A \cdot B + \overline{C}) \cdot D} = \overline{A \cdot B + \overline{C} + D} = \overline{(\overline{A + B}) \cdot C + D} \\ = (\overline{A + B}) \cdot C + \overline{D} \quad R_{11}$$

2.4 Ejercicio 3-38

3-38

Determine las condiciones de entrada necesarias para que la salida de la Figura 3-57 pase a su estado activo.



$$\overline{\overline{A}B} + \overline{\overline{B}\overline{C}} + \overline{D} + \overline{E}$$

$$\overline{A} \cdot B + \overline{B} \cdot \overline{C} + \overline{D} + \overline{E}$$

R// Las condiciones de entrada necesarias para que la salida pase a su estado activo son que se cumplan al menos una de las siguientes situaciones:

- $A = 0$ y $B = 1$
- $B = 0$ y $C = 0$
- $D = 0$
- $E = 1$

2.5 Ejercicio 4-11

4-11. Determine la expresión mínima para cada función K en la figura 4-67. Preste especial atención al paso 5 para la función en (a)

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	1	1	1	1
$\bar{A}B$	1	1	0	0
AB	0	0	0	1
$A\bar{B}$	0	0	1	1

$$(a)^* = \bar{A}\bar{B} + \bar{A}B\bar{C} + A\bar{C}\bar{D} + A\bar{B}\bar{C} //$$

(a)*

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	1	0	1	1
$\bar{A}B$	1	0	0	1
AB	0	0	0	0
$A\bar{B}$	1	0	1	1

$$(b) = \bar{A}\bar{C}\bar{D} + \bar{B}CD + \bar{A}C\bar{D} + A\bar{B}\bar{D} //$$

(b)

	\bar{C}	C
$\bar{A}\bar{B}$	1	1
$\bar{A}B$	0	0
AB	1	0
$A\bar{B}$	1	x

$$(c) = \bar{A}\bar{B} + A\bar{C} //$$

(c)

2.6 Ejercicio 4-12

4-12. Para la tabla de verdad, crea un mapa de Karnaugh 2×2 , agrupa términos y simplifica. Luego, verifica en la tabla si la expresión es verdadera para cada entrada.

A	B	Y
0	0	1
0	1	1
1	0	0
1	1	0

A \ B	0	1
0	1	1
1	0	0

$$Y = \bar{A}$$

Verificación

A	B	Y	$Y = \bar{A}$
0	0	1	1
0	1	1	1
1	0	0	0
1	1	0	0

Y coincide con el valor de $Y = \bar{A}$ en la tabla de verdad

ATTA siempre es 0 nos sigue

ATTA siempre es 0 nos sigue

3. BIBLIOGRAFÍA

- [1] C. S. Roero and E. Luciano, "Leibniz e Peano. Dalla numerazione binaria alle macchine," *Lo Sguardo*, vol. 32, 2022, doi: 10.5281/zenodo.5838199.
- [2] D. Natarajan, *Fundamentals of Digital Electronics*, vol. 623. Cham: Springer International Publishing, 2020. doi: 10.1007/978-3-030-36196-9.
- [3] W.-L. Chang and A. V. Vasilakos, *Fundamentals of Quantum Programming in IBM's Quantum Computers*, vol. 81. Cham: Springer International Publishing, 2021. doi: 10.1007/978-3-030-63583-1.
- [4] M. F. Mudawar, "Exact Versus Inexact Decimal Floating-Point Numbers and Arithmetic," *IEEE Access*, vol. 11, pp. 17891–17905, 2023, doi: 10.1109/ACCESS.2023.3244891.
- [5] J. Tian, D. W. Braithwaite, and R. S. Siegler, "Distributions of textbook problems predict student learning: Data from decimal arithmetic.," *J Educ Psychol*, vol. 113, no. 3, pp. 516–529, 2021, doi: 10.1037/edu0000618.
- [6] "Boolean Algebra and Logic Gates," in *Foundations for Fintech*, vol. Volume 1, in Global Fintech Institute - World Scientific Series on Fintech, vol. Volume 1. , WORLD SCIENTIFIC, 2021, pp. 147–160. doi: doi:10.1142/9789811238819_0007.
- [7] B. H. Suits, "Digital I," in *Electronics for Physicists: An Introduction*, B. H. Suits, Ed., Cham: Springer International Publishing, 2020, pp. 247–275. doi: 10.1007/978-3-030-39088-4_12.
- [8] J. Feng, R. Zhao, and Y. Cui, "Simplification of logical functions with application to circuits," *Electronic Research Archive*, vol. 30, no. 9, pp. 3320–3336, 2022, doi: 10.3934/era.2022168.
- [9] K. Erciyes, "Boolean Algebras and Combinational Circuits," in *Discrete Mathematics and Graph Theory: A Concise Study Companion and Guide*, K. Erciyes, Ed., Cham: Springer International Publishing, 2021, pp. 173–195. doi: 10.1007/978-3-030-61115-6_9.
- [10] E. D. Nugroho, "Development of Applications for Simplification of Boolean Functions using Quine-McCluskey Method," *Telematika*, vol. 18, no. 1, p. 27, Mar. 2021, doi: 10.31315/telematika.v18i1.3195.
- [11] S. Kurgalin and S. Borzunov, "Boolean Algebra," 2020, pp. 217–249. doi: 10.1007/978-3-030-42221-9_6.
- [12] J. Freixas, "On the enumeration of Boolean functions with distinguished variables," *Soft comput*, vol. 25, no. 19, pp. 12627–12640, Oct. 2021, doi: 10.1007/s00500-020-05422-5.
- [13] B. Steinbach and C. Posthoff, *Boolean Differential Calculus*. Cham: Springer International Publishing, 2017. doi: 10.1007/978-3-031-79892-4.
- [14] W. Jin, X. Du, J. Hu, and Y. Sun, "Boolean Functions with a Few Walsh Transform Values," 2021, pp. 642–655. doi: 10.1007/978-3-030-78618-2_53.
- [15] K. Erciyes, "Boolean Algebras and Combinational Circuits," 2021, pp. 173–195. doi: 10.1007/978-3-030-61115-6_9.

- [16] C. Carlet, "Boolean Functions," in *Encyclopedia of Cryptography, Security and Privacy*, Cham: Springer Nature Switzerland, 2025, pp. 292–296. doi: 10.1007/978-3-030-71522-9_336.
- [17] H. H. Ward, "Combinational Logic," in *Mastering Digital Electronics: An Ultimate Guide to Logic Circuits and Advanced Circuitry*, H. H. Ward, Ed., Berkeley, CA: Apress, 2024, pp. 273–324. doi: 10.1007/978-1-4842-9878-7_7.
- [18] B. J. LaMeres, *Introduction to Logic Circuits & Logic Design with VHDL*. Cham: Springer International Publishing, 2024. doi: 10.1007/978-3-031-42547-9.
- [19] B. Sampathkumar, R. Das, B. Martin, F. Enescu, and P. Kalla, "An Algebraic Approach to Partial Synthesis of Arithmetic Circuits," in *Proceedings of the Asia and South Pacific Design Automation Conference, ASP-DAC*, Institute of Electrical and Electronics Engineers Inc., Mar. 2025, pp. 1097–1103. doi: 10.1145/3658617.3697724.
- [20] D. Kaufmann, "Formal verification of multiplier circuits using computer algebra," *it - Information Technology*, vol. 64, no. 6, pp. 285–291, Dec. 2022, doi: 10.1515/itit-2022-0039.

4. ANEXO

https://github.com/Moralito64/Ejercicios_C