UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

INGENIERÍA EN COMPUTACIÓN

ESTRUCTURAS DISCRETAS

GRUPO 6

ING. ORLANDO ZALDÍVAR ZAMORATEGUI

SEMESTRE 2024-1

Sistemas algebraicos aplicados a funciones booleanas. Programa de cómputo del método de Quine-McCluskey para minimizar funciones booleanas.

EQUIPO 7
BARRIOS AGUILAR DULCE MICHELLE
DOMÍNGUEZ PALACIOS JESÚS ALEJANDRO
GARCÍA VÁZQUEZ JAVIER ALEJANDRO
MARÍN MONTAÑO JOSUÉ
PÉREZ GONZÁLEZ SHARON LESLIE

ÍNDICE

TEMA **2**

TEMARIO **2**

OBJETIVOS **2**

INTRODUCCIÓN **3**

DEFINICIONES Y CONCEPTOS **4**

SISTEMAS ALGEBRAICOS 4
ÁLGEBRA BOOLEANA 5
MÉTODOS DE MINIMIZACIÓN 6
Método algebraico *6*
Mapas de Karnaugh *8*
Método de minimización de Quine-McCluskey *8*

EJEMPLOS **13**

EJEMPLO 1 13
EJEMPLO 2 19
EJEMPLO 3 24
EJEMPLO 4 30
EJEMPLO 5 38
EJEMPLO 6 44
EJEMPLO 7 52
EJEMPLO 8 57
EJEMPLO 9 62
EJEMPLO 10 67

CUESTIONARIO **72**

BIBLIOGRAFÍA **87**

Tema

Sistemas Algebraicos aplicados a funciones booleanas. Programa de cómputo del método de Quine-McCluskey para miniminzar funciones booleanas.

Temario

1. Objetivo
2. Introducción
3. Definiciones y Conceptos.
   3.1. Sistemas Algebraicos
   3.2. Álgebra Booleana
   3.3. Métodos de minimización (menciones y breves definiciones)
   3.4. Método Quine-McCluskey
4. Ejemplos
5. Video
6. Cuestionario
7. Software/Programa
8. Bibliografía
9. Evaluación

Objetivos

1. Investigar acerca de los sistemas algebraicos aplicados en un entorno booleano, profundizando en el método de minimización de Quine-McCluskey.

2. Realizar una explicación clara para un interesado en diseñar algún circuito lógico, proporcionando los conceptos necesarios y algunos métodos de minimización para el álgebra booleana a usar, incluyendo ejemplos.

3. Proporcionar un software capaz de interpretar el método Quine-McCluskey, como método de minimización principal del proyecto, con tal de que le sea de apoyo al usuario.

Introducción

En este tutorial se presenta uno de los tres métodos de minimización aplicados a funciones booleanas, el propuesto por el filósofo Willard Van Orman Quine y el ingeniero Edward J. McCluskey; denominado método de Quine-McCluskey. Considerando: objetivos, introducción, definiciones y conceptos, ejemplos, un video, cuestionario, software y bibliografía.

Dentro de la sección de ‘definiciones y conceptos’ se proporciona la parte teórica del tema abarcando desde sistemas algebraicos, pasando por lo que es el álgebra de Boole (o álgebra booleana); ambos aspectos tratados para preparar al usuario a entender de dónde viene el entorno algebraico en que se enfocan los métodos de minimización. Este último punto, hecho para explicar a rasgos generales en lo que consiste el método algebraico, mapas de Karnaugh y en una sección por separado de los anteriores métodos; el que nos interesa, el de Quine-McCluskey.

Dentro de la sección de nuestro método, se encuentra un breve resumen de lo que se ofrece de esta forma de minimizar funciones booleanas, lo que hay que identificar de una función booleana, además de un desglose de pasos que muestran cómo se desarrolla el método. También, le adjuntamos (como bien ya fue mencionado) diez ejemplos explicados paso a paso. Pero, si la explicación dentro del desarrollo, y los diez ejemplos no son suficientes, el video que ofrecemos incluye un ejemplo desarrollado y explicado.

Por último, tenemos un cuestionario para evaluar su aprendizaje con los conceptos adquiridos con el tutorial. Y como soporte para el usuario, les adjuntamos un programa que funciona a base del método de Quine-McCluskey, que carga las tablas y muestra el resultado de la minimización.

DEFINICIONES Y CONCEPTOS

Sistemas algebraicos

Solemos llamarles sistemas algebraicos a las estructuras matemáticas que consisten en un conjunto de elementos unidos a una o más operaciones definidas. Estas operaciones son las principales que encontramos en la aritmética, como la suma y multiplicación, hasta operaciones más abstractas.

Sin embargo, usualmente se manejan datos de los tipos que representan cantidades; los números reales (π^4, $\frac{e}{2}, \sqrt{2}+1$, e, π, entre otros) en donde encontramos aún más tipos, como: los números racionales, enteros, naturales y por otro lado tenemos a los números imaginarios y luego los complejos. Mismos que van a ser representados de otra manera, en caso de desconocerlos. Por lo tanto, habrá que definir a los sistemas algebraicos como:

> Los conjuntos de una o más ecuaciones que posean más de una incógnita y que conforman un problema matemático; para el cual nos compete encontrar los valores que van a tomar las incógnitas que satisfacen las operaciones que forman parte de las ecuaciones ya mencionadas; a esto nos referimos a que también se debe cumplir la igualdad del sistema en cuestión.

Dentro de los sistemas algebraicos, encontramos ciertas representaciones de elementos en las operaciones que conforman las ecuaciones, como letras del alfabeto latino (las últimas, como: u, v, w, x, y, z) o si son demasiadas representaciones usando una misma letra, se suelen utilizar los subíndices.

Bien, conocemos lo que son los sistemas algebraicos y la definición anterior es suficiente para lo que necesitamos, pero repasemos en forma de mención a las propiedades que regulan lo que se puede o no hacer con los sistemas.

PROPIEDADES
- **Cerradura.**
- **Asociativa.**
- **Identidad.**
- **Inverso.**
- **Distributiva.**
- **Cancelación.**
- **Idempotencia.**
- **Homomorfismo (morfismo):** una función que preserva operaciones definidas en objetos matemáticos con la misma estructura algebraica.

  Ejemplo de ello: Sean $A = (A, º1, \dots, ºk)$ y $\beta = (B, º1, \dots, ºk)$ dos sistemas algebraicos de un mismo tipo, A y B son conjuntos y los elementos dentro de los paréntesis son las operaciones algebraicas definidas para esos conjuntos.

Como podemos observar, las propiedades de los sistemas algebraicos son casi los mismos que se aplican para el álgebra booleana que mencionamos en la introducción. Álgebra que, en vez de manejar una cantidad exorbitante de números, solo va a trabajar con dos, basados en el trabajo de George Boole (1815 - 1864).

Álgebra Booleana

El álgebra booleana, o también conocida como álgebra de Boole, es un sistema algebraico basado en la lógica proposicional, utilizado para representar circuitos lógicos en forma de ecuaciones. Apoyándose a su vez, de la lógica binaria y la teoría de conjuntos. Fue desarrollada por el matemático y lógico británico George Boole durante el siglo XIX.

Como ya se mencionó, en este sistema algebraico de base binaria, nos indica que se van a manipular valores binarios, esto nos dice que son valores que pueden ser verdaderos (1) o falsos (0). Y al ser un sistema algebraico, se van a estipular ciertos reglamentos para su adecuado y lógico uso, utilizando operaciones como conjunción (AND), disyunción (OR) y la negación (NOT). Nos permite evaluar condiciones, tomar decisiones y controlar el flujo de ejecución de programas que queramos poner en marcha.

APLICACIONES DEL ÁLGEBRA DE BOOLE:
- Diseño de circuitos lógicos (digitales: procesadores, memorias y dispositivos lógicos programables).
- Programación y algoritmos (estructuras de control condicional).
- Diseño de hardware.

NOS PERMITE:
- Debido a sus leyes, simplificar expresiones booleanas complejas, facilitando su comprensión y análisis. (Como lo hemos visto en los distintos métodos de minimización, y ahora con Quine-McCluskey).
- Optimizar circuitos digitales. (Se reducen el número de compuertas lógicas, mejorando su eficacia.)

Y proporciona un marco matemático adecuado para el estudio de la lógica y el razonamiento lógico; asunto vital para la computación moderna.

LEYES
1. **Ley Conmutativa** – Para la operación disyuntiva (OR) nos dice que el orden que ocupen las variables es indiferente. Lo representamos con como: $\vee$.

2. **Ley Asociativa** – Para la operación conjuntiva (AND) el orden que ocupen las variables es indiferente, Lo representamos como: $\wedge$.

3. **Ley Distributiva** – Para la operación disyuntiva (OR), en su resultado al tener un caso en que intervienen más de dos variables, va a ser independiente del modo en que se agrupen las variables.

Métodos de minimización

Ahora estamos teniendo un enfoque centrado a la búsqueda de representaciones algebraicas de lógica para representar desde circuitos hasta estructuras condicionales que por ende nos deja en un entorno de lógica con razón de dos respuestas en las que encontramos a verdadero (1) y falso (0). Por lo que encontraremos sistemas algebraicos en los que tendremos expresiones extensas y que complicarían la comprensión de sí mismas para uno. Por lo tanto, nos quedamos en la necesidad de recurrir a métodos que logren reducir las expresiones en álgebra de Boole, que ahora; por comodidad, vamos a llamar funciones booleanas.

Para nuestro curso hemos visto distintas formas de reducir una función booleana, siendo estas las más destacadas de las que se encuentran en el entorno computacional, en el área de lógica se refiere, las cuales serían reducciones por:

- Método algebraico
- Mapas de Karnaugh.
- Método de Quine-McCluskey

Claro, el criterio de minimización más usado es obteniendo una expresión en forma de suma de productos, que tenga un número mínimo de términos con el menor número de variables posible en cada uno de estos productos. Pero para ello, se ha de obtener la expresión en su forma 'canónica', como se hace referencia en algunos textos (Como en que consultamos: SISTEMAS ELECTRONICOS DIGITALES, de 1998). Ahora si podemos hablar del primer método de minimización.

Método algebraico

Para este método es a primera instancia, fácil de interpretar por su semejanza con los sistemas algebraicos a los que estamos acostumbrados. Sin embargo, aquí iremos aplicando los postulados (o propiedades) del álgebra de Boole.

Bueno, las propiedades que se estuvieron manejando durante el curso para la aplicación de este método se encuentran en la tabla VI que adjuntamos a continuación, en donde estamos manejando las representaciones de la conjunción, disyunción y negación en su forma (· , + y ', respectivamente).

Expresión	Equivalencia, ley o propiedad
p’’ = p	Doble negación
p + p = p p · p = p	Idempotencia
p + q = q + p p · q = q · p	Conmutativa
p + (q + r) = (p + q) + r p · (q · r) = (p · q) · r	Asociativa
p + (q · r) = (p + q) · (p + r) p · (q + r) = (p · q) +(p · r)	Distributiva
p + p’ = 1 p · p’ = 0	Tercero exculido Complemento Tautología Tercero exculido Complemento Contradicción
p + 0 = p p · 1 = p	Identidad
p + 1 = 1 p · 0 = 0	Dominancia o Dominante
(p + q)’ = p’ · q’ (p · q)’ = p’ + q’	De Morgan
p + (p · q) = p p · (p + q) = p	Absorción

Tabla VI. Leyes y propiedades (algebra booleana)
1 corresponde a True
0 corresponde a False

Mapas de Karnaugh

Este método surge de tratar de agrupar a los términos de interés sin aplicar de forma directa el desarrollo algebraico que ofrece el método anterior. Y esto se constituye de desarrollar de forma gráfica la tabla de verdad de una función lógica.

Como podemos ver, se pueden tratar funciones de dos hasta cinco variables, para este método tabular cuyos términos canónicos adyacentes pueden ser agrupados para hacer de forma más sencilla la agrupación de los elementos.

Su algoritmo, en pocas palabras, consiste en:

1. Se escribe un 1 en los cuadros que tiene un número de los que forman la sumatoria de la función.
2. Se forman los grupos de minitérminos que se consideran vecinos (tienen un lado en común).
3. Se identifican a las variables para los conjuntos hechos por los grupos que hicimos de los minitérminos, estos serán los conjuntos.
4. Hacemos la sumatoria de los productos *Habrá que comprobar el resultado con el método gráfico de formar líneas dictadas por las variables que forman a los productos de los grupos*.

Método de minimización de Quine-McCluskey

Este método fue presentado originalmente por Willard Van Orman Quine y Edward J. McCluskey, denominado en su momento como método (Q-M), no limitado al número de variables de una función (como si lo pudiera llegar a ser para los mapas de Karnaugh), posibilita la implementación algorítmica y tiene la ventaja de realizar simplificaciones simultáneas para varias funciones que comparten un mismo conjunto de variables de entrada. Entrega una solución minimizada global de la implementación, en vez de una solución mínima por función (los llamados mínimos locales).

A considerar que la función booleana como:

variables Minitérminos

$$f\boxed{(a,b,c,d)} = \sum_m \boxed{(0,1,2,3,4)}$$

Sumatoria de minitérminos

Se nos puede presentar una función booleana de:

- 2 variables: $f(a,b)$
- 3 variables: $f(a,b,c)$
- 4 variables: $f(a,b,c,d)$
- O hasta 7 variables: $f(a,b,c,d,e,f,g)$

Y eso nos va a indicar que las representaciones binarias de los minitérminos podrían llegar a representarse dependiendo de la cantidad de variables presentes en la función boolena:

Cantidad de variables	Minitérminos	Representaciones por esperar
2	0 1 2	00 01 11
3	0 1 2 3 4 5 6 7	000 001 010 011 100 101 110 111
4	0 1 2 3	0000 0001 0010 0011

Y así sucesivamente.
Entonces, para su desarrollo se realiza por medio de la elaboración de cinco tablas en las que se van a ir agregando datos específicos, así que, para minimizar una función booleana se podría resumir en:

1. Elaborar una tabla 1 en donde vamos a ir almacenando a los minitérminos de la función en la primera columna, su representación binaria en la tercera columna, y por último la segunda columna la cantidad de 1's en su representación binaria (0, 1, 2, 3, …, n).

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	
1	1	0001	
2	1	0010	
3	2	0011	
4	1	0100	

***NOTA: la cuarta columna sirve de apoyo para agregar comentarios al momento de ejercer algún movimiento importante en el desarrollo y es necesario volver a checar qué se hizo. Aunque, nosotros lo hemos trabajado como una columna para escribir una marca de tipo *, que vamos a explicar en un momento.**

2. Generamos una tabla 2, en esta vamos a reordenar a los minitérminos y su representación de acuerdo con la cantidad de 1's que tienen en su representación.

Tabla 2

Minitérminos	abcd	1's	
0	0000	0	
1 2 4	0001 0010 0100	1	
3	0011	2	

Y vamos marcando con un * en la tabla 1, a los minitérminos que ya hemos agregado a la tabla 2.

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
1	1	0001	*
2	1	0010	*
3	2	0011	*
4	1	0100	*

3. Generamos una tabla 3, que nos servirá para empezar a realizar combinaciones. Efecto que puede ser presentado una vez que se pase por los siguientes puntos:

   i. Buscamos en la representación binaria de los minitérminos, los que difieren en solo una variable (la que sea) para poder realizar una combinación con apoyo en la tabla 2.
   ii. Empezamos con el primer caso en la cantidad de 1's con la siguiente cantidad. En este caso los que tienen cero 1's con los que tiene un solo 1.
   iii. Al combinar a los minitérminos, si tenemos:
   0 y 0 escribimos 0
   1 y 1 escribimos 1
   0 y 1 escribimos -

Tabla 3

Combinación	abcd	1's	
(0,1) (0,2) (0,4)	000- 00-0 0-00	0	
(1,3)	00-1	1	

(2,3)	001-		

Y vamos marcando en la tabla 2 a los minitérminos que vamos combinando.

Tabla 2

Minitérminos	**abcd**	**1's**	
0	0000	0	*
1 2 4	0001 0010 0100	1	* * *
3	0011	2	*

NOTA: Puede haber casos en los que nos quede un minitérmino que no se pueda combinar con los otros que están involucrados en la función booleana. Estos ya son considerados implicantes primos y van directo a la tabla del mismo nombre.

4. Generamos una tabla 4, para realizar una segunda combinación; sin embargo, serán combinaciones de combinaciones, apoyándonos en la tabla anterior (la tabla 3).

   Para esta segunda vuelta de combinaciones, nos seguimos basando en las reglas de combinación que hemos descrito anteriormente.

Tabla 4

Combinación	**abcd**	**1's**	
(0,1,2,3)	00--	0	

Tabla 3

Combinación	**abcd**	**1's**	
(0,1) (0,2) (0,4)	000- 00-0 0-00	0	* *
(1,3) (2,3)	00-1 001-	1	* *

Para la combinación de combinaciones vamos a considerar que sólo una variable difiera entre las representaciones binarias de las combinaciones. Por ejemplo: (0,1) combinado con (2,3) cuyas representaciones son 000- y 001- respectivamente, y la única variable que difiere es 'c' por lo tanto, se pueden combinar estas combinaciones.

Y así, la primera combinación de combinaciones generada sería (0,1,2,3), cuya representación en la columna abcd queda como: 00--.

Marcamos un * en la última columna indicando qué combinaciones han logrado pasar por una segunda combinación.

En caso de tener otras combinaciones que al combinarse generen una combinación igual a una ya obtenida, se van a marcar también. En este caso, con las combinaciones (0,2) y (1,3) podemos marcarlas también debido a que al combinarlas obtenemos (0,1,2,3), con una representación exactamente igual: 00-- como con la combinación anterior.

5. Generamos la tabla 5 (la tabla de los implicantes primos), y vamos almacenando a los minitérminos y combinaciones que hemos considerado como implicantes primos, además de sus representaciones binarias en su correspondiente columna y marcar con una X en las celdas que denotan un minitérmino de la función.

   Y consideramos para la escritura del producto:
   Si en la representación binaria (abcd) tenemos para el producto:
   - no hay variable representativa
   0 la variable está negada
   1 la variable no está negada

   Tabla 5

Combinación	**abcd**		**1's**		**0**	**1**	**2**	**3**	**4**	**Producto**
(0,1,2,3)	00--		0		X	X	X	X		a'b'
(0,4)	0-00		0		X				X	a'c'd'

6. Y escribimos el resultado como la suma de productos (aunque esta no es la única forma).

$$f(a,b,c,d) = \sum_m (0,1,2,3,4) = a'b' + a'c'd'$$

EJEMPLOS

A continuación, incluiremos diez ejemplos con explicación escrita paso a paso para un mejor entendimiento de la implementación de este método de minimización, con base en la forma vista en clase; tratando de abordar todos los casos posibles en funciones de cuatro a cinco variables.

EJEMPLO 1

$$f(a,b,c,d) = \sum_m (0,2,4,5,6,8)$$

Solución:

1) Generamos la primera tabla. **Objetivo:** Obtener las formas binarias de los minitérminos y la cantidad de 1's en ellos.

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	
2	1	0010	
4	1	0100	
5	2	0101	
6	2	0110	
8	1	1000	

- **Columna minitérminos:** llenamos conforme a los minitérminos que nos muestra la función en este caso: 0, 2, 4, 5, 6 y 8
- **Columna 1´s:** es el número de **1´s** que contiene el número binario de ese minitérmino
- **abcd:** el número binario que representa ese minitérmino

2) Generamos la segunda tabla. **Objetivo:** Reagrupar a los minitérminos con base en la cantidad de 1's en su representación binaria.

PASO 2.1

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	
4	1	0100	
5	2	0101	
6	2	0110	
8	1	1000	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	

Marcamos con un * en la Tabla 1, para indicar que hemos agregado los minitérminos en la Tabla 2.

PASO 2.2

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
4	1	0100	
5	2	0101	
6	2	0110	
8	1	1000	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2	0010		1	

PASO 2.3

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
4	1	0100	*
5	2	0101	
6	2	0110	
8	1	1000	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2 4	0010 0100		1	

PASO 2.4

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
4	1	0100	*
5	2	0101	*
6	2	0110	
8	1	1000	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2 4 8	0010 0100 1000		1	

PASO 2.5

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
4	1	0100	*
5	2	0101	*
6	2	0110	*
8	1	1000	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2 4 8	0010 0100 1000		1	
5	0101		2	

PASO 2.6

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
4	1	0100	*
5	2	0101	*
6	2	0110	*
8	1	1000	*

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2 4 8	0010 0100 1000		1	
5 6	0101 0110		2	

3) Generamos una tercera tabla. **Objetivo:** Realizar la primera combinación con minitérminos.
**Pasamos a la parte de realización de combinaciones con los minitérminos.

COMBINACIONES:

1. Buscamos en la representación binaria de los minitérminos, los que difieren en solo una variable (la que sea) para poder realizar una combinación con apoyo en la tabla 2.
2. Empezamos con el primer caso en la cantidad de 1's con la siguiente cantidad. En este caso los que tienen cero 1's con los que tiene un solo 1.
3. Al combinar a los minitérminos, si tenemos:
   0 y 0 será 0
   1 y 1 será 1
   0 y 1 será -

Ejemplo de una combinación de minitérminos:

Buscamos combinar al minitérmino 0 y 2.

Minitérmino	abcd
0	0000
2	0010

COMBINACIÓN: 00–0

PASO 3.1

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
2 4 8	0010 0100 1000	*	1	
5 6	0101 0110		2	

Tabla 3

Combinación	abcd		1's	
(0,2)	00-0		0	

Marcamos con un * en la Tabla 2, para indicar que hemos agregado las combinaciones de minitérminos posibles en la Tabla 3.

PASO 3.2

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
2 4 8	0010 0100 1000	* *	1	
5 6	0101 0110		2	

Tabla 3

Combinación	abcd		1's	
(0,2) (0,4)	00-0 0-00		0	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

PASO 3.3

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
2 4 8	0010 0100 1000	* * *	1	
5 6	0101 0110		2	

Tabla 3

Combinación	abcd		1's	
(0,2) (0,4) (0,8)	00-0 0-00 -000		0	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

PASO 3.4

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
2 4 8	0010 0100 1000	* * *	1	
5 6	0101 0110	*	2	

Tabla 3

Combinación	abcd		1's	
(0,2) (0,4) (0,8)	00-0 0-00 -000		0	
(4,5)	010-		1	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

PASO 3.5

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
2 4 8	0010 0100 1000	* * *	1	
5 6	0101 0110	* *	2	

Tabla 3

Combinación	abcd		1's	
(0,2) (0,4) (0,8)	00-0 0-00 -000		0	
(4,5) (4,6)	010- 01-0		1	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

4) Generamos una cuarta tabla. **Objetivo:** Realizar la segunda combinación con las combinaciones de la tabla anterior.

Buscamos combinar a la combinación (0,2) y (4,6).

Combinación	abcd
(0,2)	00-0
(4,6)	01-0

COMBINACIÓN: 0—0

Tabla 3

Combinación	abcd		1's	
(0,2) (0,4) (0,8)	00-0 0-00 -000	* * 	0	
(4,5) (4,6)	010- 01-0	 *	1	

Tabla 4

Combinación	abcd		1's	
(0,2,4,6)	0—0		0	

Marcamos con un * para las combinaciones (0,2) y (4,6) primero por la combinación que estamos realizando. Sin embargo, como tenemos representado también a (0,4), lo marcaremos de igual manera en la Tabla 3.

5) Generamos la tabla 5. **Objetivo:** Colocar a los implicantes primos (combinaciones que ya no pueden ser combinadas con otras; se denotan por no tener una marca * en las tablas 3 y 4), en una sola tabla para obtener a los productos.

Tabla 5. Implicantes primos

Combinación	abcd		1's		0	2	4	5	6	8	Producto
(0,2,4,6)	0—0		0		X	X	X		X		a'd'
(0,8)	-000		0		X					X	b'c'd'
(4,5)	010-		1				X	X			a'bc'

NOTA: Si en la representación binaria (abcd) tenemos para el producto:
- no hay variable representativa
0 la variable está negada
1 la variable no está negada

FINALMENTE OBTENEMOS LA MINIMIZACIÓN DADA POR LA SUMA DE LOS PRODUCTOS, resultado que podemos representar como:

$$f(a,b,c,d) = \sum\nolimits_m (0,2,4,5,6,8) = a'd' + b'c'd' + a'bc'$$

O simplemente:

$$f(a,b,c,d) = a'd' + b'c'd' + a'bc'$$

EJEMPLO 2

$$f(a,b,c,d) = \sum_m (1,3,9,11)$$

Solución:

1) Generamos la primera tabla. **Objetivo:** Obtener las formas binarias de los minitérminos y la cantidad de 1's en ellos.

Tabla 1

Minitérminos	1's	abcd	
1	1	0001	
3	2	0011	
9	2	1001	
11	3	1011	

- **Columna minitérminos:** llenamos conforme a los minitérminos que nos muestra la función en este caso: 1, 3, 9, 11.
- **Columna 1´s:** es el número de **1´s** que contiene el número binario de ese minitérmino
- **abcd:** el número binario que representa ese minitérmino

2) Generamos la segunda tabla. **Objetivo:** Reagrupar a los minitérminos con base en la cantidad de 1's en su representación binaria.

PASO 2.1

Tabla 1

Minitérminos	1's	abcd	
1	1	0001	*
3	2	0011	
9	2	1001	
11	3	1011	

Tabla 2

Minitérminos	abcd		1's	
1	0001		1	

Marcamos con un * en la Tabla 1, para indicar que hemos agregado los minitérminos en la Tabla 2.

PASO 2.2

Tabla 1

Minitérminos	1's	abcd	
1	1	0001	*
3	2	0011	*
9	2	1001	
11	3	1011	

Tabla 2

Minitérminos	abcd		1's	
1	0001		1	
3	0011		2	

PASO 2.3

Tabla 1

Minitérminos	1's	abcd	
1	1	0001	*
3	2	0011	*
9	2	1001	*
11	3	1011	

Tabla 2

Minitérminos	abcd		1's	
1	0001		1	
3 9	0011 0101		2	

PASO 2.4

Tabla 1

Minitérminos	1's	abcd	
1	1	0001	*
3	2	0011	*
9	2	1001	*
11	3	1011	*

Tabla 2

Minitérminos	abcd		1's	
1	0001		1	
3 9	0011 0101		2	
11	1011		3	

3) Generamos una tercera tabla. **Objetivo:** Realizar la primera combinación con minitérminos.
**Pasamos a la parte de realización de combinaciones con los minitérminos.

COMBINACIONES:

1. Buscamos en la representación binaria de los minitérminos, los que difieren en solo una variable (la que sea) para poder realizar una combinación con apoyo en la tabla 2.
2. Empezamos con el primer caso en la cantidad de 1's con la siguiente cantidad. En este caso los que tienen cero 1's con los que tiene un solo 1.
3. Al combinar a los minitérminos, si tenemos:
   0 y 0 será 0
   1 y 1 será 1
   0 y 1 será -

Ejemplo de una combinación de minitérminos:
Buscamos combinar al minitérmino 0 y 2.

Minitérmino	abcd
1	0001
3	0011

COMBINACIÓN: 00–1

PASO 3.1

Tabla 2

Minitérminos	abcd		1's	
1	0001	*	1	
3 9	0011 1001	*	2	
11	1011		3	

Tabla 3

Combinación	abcd		1's	
(1,3)	00-1		1	

Marcamos con un * en la Tabla 2, para indicar que hemos agregado las combinaciones de minitérminos posibles en la Tabla 3.

PASO 3.2

Tabla 2

Minitérminos	abcd		1's	
1	0001	*	1	
3 9	0011 1001	* *	2	
11	1011		3	

Tabla 3

Combinación	abcd		1's	
(1,3) (1,9)	00-1 -001		1	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

PASO 3.3

Tabla 2

Minitérminos	abcd		1's	
1	0001	*	1	
3 9	0011 1001	* *	2	
11	1011	*	3	

Tabla 3

Combinación	abcd		1's	
(1,3) (1,9)	00-1 -001		1	
(3,11) (9,11)	-011 10-1		2	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

4) Generamos una cuarta tabla. **Objetivo:** Realizar la segunda combinación con las combinaciones de la tabla anterior.

Buscamos combinar a las combinaciones:

Combinación	abcd
(1,3)	00-1
(9,11)	10-1

COMBINACIÓN: -0-1

Tabla 3

Combinación	abcd		1's	
(1,3) (1,9)	00-1 -001	* *	1	
(3,11) (9,11)	-011 10-1	* *	2	

Tabla 4

Combinación	abcd		1's	
(1,3,9,11)	-0-1		1	

Marcamos con un * para las combinaciones (1,3), (9,11) que se están combinando por el ejemplo visto arriba. Sin embargo, vemos que al combinar (1,9) y (3,11) obtenemos -0-1 igual que la primera combinación mencionada. Por lo tanto, las marcamos también.

5) Generamos la tabla 5. **Objetivo:** Colocar a los implicantes primos (combinaciones que ya no pueden ser combinadas con otras; se denotan por no tener una marca * en las tablas 3 y 4), en una sola tabla para obtener a los productos.

Tabla 5. Implicantes primos

Combinación	abcd		1's		1	3	9	11	Producto
(1,3,9,11)	-0-1		1		X	X	X	X	b'd

NOTA: Si en la representación binaria (abcd) tenemos para el producto:
- no hay variable representativa
0 la variable está negada
1 la variable no está negada

FINALMENTE OBTENEMOS LA MINIMIZACIÓN DADA POR LA SUMA DE LOS PRODUCTOS, resultado que podemos representar como:

$$f(a,b,c,d) = \sum\nolimits_m (1,3,9,11) = b'd$$

O simplemente:

$$f(a,b,c,d) = b'd$$

EJEMPLO 3

$$f(a,b,c,d) = \sum\nolimits_m (0,1,4,5,11,15)$$

Solución:

1) Generamos la primera tabla. **Objetivo:** Obtener las formas binarias de los minitérminos y la cantidad de 1's en ellos.

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	
1	1	0010	
4	1	0100	
5	2	0101	
11	3	1011	
15	4	1111	

- **Columna minitérminos:** llenamos conforme a los minitérminos que nos muestra la función en este caso: 0, 1, 4, 5, 11, 15.
- **Columna 1´s:** es el número de **1´s** que contiene el número binario de ese minitérmino
- **abcd:** el número binario que representa ese minitérmino

2) Generamos la segunda tabla. **Objetivo:** Reagrupar a los minitérminos con base en la cantidad de 1's en su representación binaria.

PASO 2.1

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
1	1	0010	
4	1	0100	
5	2	0101	
11	3	1011	
15	4	1111	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	

Marcamos con un * en la Tabla 1, para indicar que hemos agregado los minitérminos en la Tabla 2.

PASO 2.2

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
1	1	0010	*
4	1	0100	
5	2	0101	
11	3	1011	
15	4	1111	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
1	0001		1	

PASO 2.3

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
1	1	0010	*
4	1	0100	*
5	2	0101	
11	3	1011	
15	4	1111	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
1 4	0010 0100		1	

PASO 2.4

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
1	1	0010	*
4	1	0100	*
5	2	0101	*
11	3	1011	
15	4	1111	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
1 4	0010 0100		1	
5	0101		2	

PASO 2.6

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
1	1	0010	*
4	1	0100	*
5	2	0101	*
11	3	1011	*
15	4	1111	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
1 4	0010 0100		1	
5	0101		2	
11	1011		3	

PASO 2.6

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
1	1	0010	*
4	1	0100	*
5	2	0101	*
11	3	1011	*
15	4	1111	*

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
1 4	0010 0100		1	
5	0101		2	
11	1011		3	
15	1111		4	

3) Generamos una tercera tabla. **Objetivo:** Realizar la primera combinación con minitérminos.
   **Pasamos a la parte de realización de combinaciones con los minitérminos.

   COMBINACIONES:
   1. Buscamos en la representación binaria de los minitérminos, los que difieren en solo una variable (la que sea) para poder realizar una combinación con apoyo en la tabla 2.
   2. Empezamos con el primer caso en la cantidad de 1's con la siguiente cantidad. En este caso los que tienen cero 1's con los que tiene un solo 1.
   3. Al combinar a los minitérminos, si tenemos:
      0 y 0 será 0
      1 y 1 será 1
      0 y 1 será

Ejemplo de una combinación de minitérminos:
Buscamos combinar al minitérmino 0 y 2.

Minitérmino	abcd
0	0000
1	0001

COMBINACIÓN: 000–

PASO 3.1

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
1 4	0010 0100	*	1	
5	0101		2	
11	1011		3	
15	1111		4	

Tabla 3

Combinación	abcd		1's	
(0,1)	000-		0	

Marcamos con un * en la Tabla 2, para indicar que hemos agregado las combinaciones de minitérminos posibles en la Tabla 3.

PASO 3.2

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
1 4	0010 0100	* *	1	
5	0101		2	
11	1011		3	
15	1111		4	

Tabla 3

Combinación	abcd		1's	
(0,1) (0,4)	000- 0-00		0	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

PASO 3.3

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
1 4	0010 0100	* *	1	
5	0101	*	2	
11	1011		3	
15	1111		4	

Tabla 3

Combinación	abcd		1's	
(0,1) (0,4)	000- 0-00		0	
(4,5)	010-		1	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

PASO 3.4

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
1 4	0010 0100	* *	1	
5	0101	*	2	
11	1011	*	3	
15	1111	*	4	

Tabla 3

Combinación	abcd		1's	
(0,1) (0,4)	000- 0-00		0	
(4,5)	010-		1	
(11,15)	1-11		3	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

4) Generamos una cuarta tabla. **Objetivo:** Realizar la segunda combinación con las combinaciones de la tabla anterior.

Buscamos combinar a las combinaciones:

Combinación	abcd
(0,1)	000-
(4,5)	010-

COMBINACIÓN: 0-0-

Tabla 3

Combinación	abcd		1's	
(0,1) (0,4)	000- 0-00	* *	0	
(4,5)	010-	*	1	
(11,15)	1-11		3	

Tabla 4

Combinación	abcd		1's	
(0,1,4,5)	0–0-		0	

Marcamos con un * para las combinaciones (0,1) y (4,5); y también a la (0,4) por estar implicada en la combinación de combinaciones (0,1,4,5). Dejándonos con (11,15) como combinación implicante primo.

5) Generamos la tabla 5. **Objetivo:** Colocar a los implicantes primos (combinaciones que ya no pueden ser combinadas con otras; se denotan por no tener una marca * en las tablas 3 y 4), en una sola tabla para obtener a los productos.

Tabla 5. Implicantes primos

Combinación	abcd		1's		0	1	4	5	11	15	Producto
(0,1,4,5)	0–0-		0		X	X	X	X			a'c'
(11,15)	1-11		1						X	X	acd

NOTA: Si en la representación binaria (abcd) tenemos para el producto:
- no hay variable representativa
0 la variable está negada
1 la variable no está negada

FINALMENTE OBTENEMOS LA MINIMIZACIÓN DADA POR LA SUMA DE LOS PRODUCTOS, resultado que podemos representar como:

$$f(a,b,c,d) = \sum_m (0,1,4,5,11,15) = a'c' + acd$$

O simplemente:

$$f(a,b,c,d) = a'c' + acd$$

EJEMPLO 4

$$f(a,b,c,d) = \sum\nolimits_m (0,2,4,6,8,11,15)$$

Solución:

1) Generamos la primera tabla. **Objetivo:** Obtener las formas binarias de los minitérminos y la cantidad de 1's en ellos.

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	
2	1	0010	
4	1	0100	
6	2	0110	
8	1	1000	
11	3	1011	
15	4	1111	

- **Columna minitérminos:** llenamos conforme a los minitérminos que nos muestra la función en este caso: 0, 2, 4, 5, 6, 8, 11 y 15.
- **Columna 1´s:** es el número de **1´s** que contiene el número binario de ese minitérmino
- **abcd:** el número binario que representa ese minitérmino

2) Generamos la segunda tabla. **Objetivo:** Reagrupar a los minitérminos con base en la cantidad de 1's en su representación binaria.

PASO 2.1

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	
4	1	0100	
6	2	0110	
8	1	1000	
11	3	1011	
15	4	1111	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	

Marcamos con un * en la Tabla 1, para indicar que hemos agregado los minitérminos en la Tabla 2.

PASO 2.2

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
4	1	0100	
6	2	0110	
8	1	1000	
11	3	1011	
15	4	1111	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2	0010		1	

PASO 2.3

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
4	1	0100	*
6	2	0110	
8	1	1000	
11	3	1011	
15	4	1111	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2 4	0010 0100		1	

PASO 2.4

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
4	1	0100	*
6	2	0110	*
8	1	1000	
11	3	1011	
15	4	1111	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2 4	0010 0100		1	
6	0110		2	

PASO 2.5

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
4	1	0100	*
6	2	0110	*
8	1	1000	*
11	3	1011	
15	4	1111	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2 4 8	0010 0100 1000		1	
6	0110		2	

PASO 2.6

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
4	1	0100	*
6	2	0110	*
8	1	1000	*
11	3	1011	*
15	4	1111	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2 4 8	0010 0100 1000		1	
6	0110		2	
11	1011		3	

PASO 2.7

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
4	1	0100	*
6	2	0110	*
8	1	1000	*
11	3	1011	*
15	4	1111	*

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2 4 8	0010 0100 1000		1	
6	0110		2	
11	1011		3	
15	1111		4	

3) Generamos una tercera tabla. **Objetivo:** Realizar la primera combinación con minitérminos.
   **Pasamos a la parte de realización de combinaciones con los minitérminos.

   COMBINACIONES:
   1. Buscamos en la representación binaria de los minitérminos, los que difieren en solo una variable (la que sea) para poder realizar una combinación con apoyo en la tabla 2.
   2. Empezamos con el primer caso en la cantidad de 1's con la siguiente cantidad. En este caso los que tienen cero 1's con los que tiene un solo 1.
   3. Al combinar a los minitérminos, si tenemos:
      0 y 0 será 0
      1 y 1 será 1
      0 y 1 será -

Ejemplo de una combinación de minitérminos:
Buscamos combinar al minitérmino 0 y 2.

Minitérmino	abcd
0	0000
2	0010

COMBINACIÓN: 00–0

PASO 3.1

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
2 4 8	0010 0100 1000	*	1	
6	0110		2	
11	1011		3	
15	1111		4	

Tabla 3

Combinación	abcd		1's	
(0,2)	00-0		0	

Marcamos con un * en la Tabla 2, para indicar que hemos agregado las combinaciones de minitérminos posibles en la Tabla 3.

PASO 3.2

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
2 4 8	0010 0100 1000	* *	1	
6	0110		2	
11	1011		3	
15	1111		4	

Tabla 3

Combinación	abcd		1's	
(0,2) (0,4)	00-0 0-00		0	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

PASO 3.3

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
2 4 8	0010 0100 1000	* * *	1	
6	0110		2	
11	1011		3	
15	1111		4	

Tabla 3

Combinación	abcd		1's	
(0,2) (0,4) (0,8)	00-0 0-00 -000		0	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

PASO 3.4

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
2 4 8	0010 0100 1000	* * *	1	
6	0110	*	2	
11	1011		3	
15	1111		4	

Tabla 3

Combinación	abcd		1's	
(0,2) (0,4) (0,8)	00-0 0-00 -000		0	
(2,6)	0-10		1	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están

combinando.

PASO 3.5

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
2	0010	*	1	
4	0100	*		
8	1000	*		
6	0110	*	2	
11	1011		3	
15	1111		4	

Tabla 3

Combinación	abcd		1's	
(0,2)	00-0		0	
(0,4)	0-00			
(0,8)	-000			
(2,6)	0-10		1	
(4,6)	01-0			

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

PASO 3.6

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
2	0010	*	1	
4	0100	*		
8	1000	*		
6	0110	*	2	
11	1011	*	3	
15	1111	*	4	

Tabla 3

Combinación	abcd		1's	
(0,2)	00-0		0	
(0,4)	0-00			
(0,8)	-000			
(2,6)	0-10		1	
(4,6)	01-0			
(11,15)	1-11		3	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

4) Generamos una cuarta tabla. **Objetivo:** Realizar la segunda combinación con las combinaciones de la tabla anterior.

Buscamos combinar a las combinaciones:

Combinación	abcd
(0,2)	00-0
(4,6)	01-0

COMBINACIÓN: 0—0

Tabla 3

Combinación	abcd		1's	
(0,2) (0,4) (0,8)	00-0 0-00 -000	* *	0	
(2,6) (4,6)	0-10 01-0	* *	1	
(11,15)	1-11		3	

Tabla 4

Combinación	abcd		1's	
(0,2,4,6)	0—0		0	

Marcamos con un * para las combinaciones (0,2), (0,4), (2,4) y (4,6), porque sus combinaciones son iguales a (0,2,4,6).

5) Generamos la tabla 5. **Objetivo:** Colocar a los implicantes primos (combinaciones que ya no pueden ser combinadas con otras; se denotan por no tener una marca * en las tablas 3 y 4), en una sola tabla para obtener a los productos.

Tabla 5. Implicantes primos

Combinación	abcd		1's		0	2	4	6	8	11	15	Producto
(0,2,4,6)	0—0		0		X	X	X	X				a'd'
(0,8)	-000		0		X				X			b'c'd'
(11,15)	1-11		3							X	X	acd

NOTA: Si en la representación binaria (abcd) tenemos para el producto:
- no hay variable representativa
0 la variable está negada
1 la variable no está negada

FINALMENTE OBTENEMOS LA MINIMIZACIÓN DADA POR LA SUMA DE LOS PRODUCTOS, resultado que podemos representar como:

$$f(a,b,c,d) = \sum\nolimits_{m} (0,2,4,6,8,11,15) = a'd' + b'c'd' + abc$$

O simplemente:

$$f(a,b,c,d) = a'd' + b'c'd' + abc$$

EJEMPLO 5

$$f(a,b,c,d) = \sum\nolimits_m (0,2,4,6,9)$$

Solución:

1) Generamos la primera tabla. **Objetivo:** Obtener las formas binarias de los minitérminos y la cantidad de 1's en ellos.

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	
2	1	0010	
4	1	0100	
6	2	0110	
9	2	1001	

- **Columna minitérminos:** llenamos conforme a los minitérminos que nos muestra la función en este caso: 0, 2, 4, 6 y 9
- **Columna 1´s:** es el número de **1´s** que contiene el número binario de ese minitérmino
- **abcd:** el número binario que representa ese minitérmino

2) Generamos la segunda tabla. **Objetivo:** Reagrupar a los minitérminos con base en la cantidad de 1's en su representación binaria.

PASO 2.1

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	
4	1	0100	
6	2	0110	
9	2	1001	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	

Marcamos con un * en la Tabla 1, para indicar que hemos agregado los minitérminos en la Tabla 2.

PASO 2.2

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
4	1	0100	
6	2	0110	
9	2	1001	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2	0010		1	

PASO 2.3

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
4	1	0100	*
6	2	0110	
9	2	1001	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2 4	0010 0100		1	

PASO 2.4

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
4	1	0100	*
6	2	0110	*
9	2	1001	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2 4	0010 0100		1	
6	0110		2	

PASO 2.5

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
4	1	0100	*
6	2	0110	*
9	2	1001	*

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2 4	0010 0100		1	
6 9	0110 1001		2	

3) Generamos una tercera tabla. **Objetivo:** Realizar la primera combinación con minitérminos.
**Pasamos a la parte de realización de combinaciones con los minitérminos.

COMBINACIONES:

1. Buscamos en la representación binaria de los minitérminos, los que difieren en solo una variable (la que sea) para poder realizar una combinación con apoyo en la tabla 2.
2. Empezamos con el primer caso en la cantidad de 1's con la siguiente cantidad. En este caso los que tienen cero 1's con los que tiene un solo 1.
3. Al combinar a los minitérminos, si tenemos:
   0 y 0 será 0
   1 y 1 será 1
   0 y 1 será -

Ejemplo de una combinación de minitérminos:

Buscamos combinar al minitérmino 0 y 2.

Minitérmino	abcd
0	0000
2	0010

COMBINACIÓN: 00–0

PASO 3.1

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
2 4	0010 0100	*	1	
6 9	0110 1001		2	

Tabla 3

Combinación	abcd		1's	
(0,2)	00-0		0	

Marcamos con un * en la Tabla 2, para indicar que hemos agregado las combinaciones de minitérminos posibles en la Tabla 3.

PASO 3.2

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
2 4	0010 0100	* *	1	
6 9	0110 1001		2	

Tabla 3

Combinación	abcd		1's	
(0,2) (0,4)	00-0 0-00		0	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

PASO 3.4

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
2 4	0010 0100	* *	1	
6 9	0110 1001	*	2	

Tabla 3

Combinación	abcd		1's	
(0,2) (0,4)	00-0 0-00		0	
(2,6)	0-10		1	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

PASO 3.5

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
2 4	0010 0100	* *	1	
6 9	0110 1001	*	2	

Tabla 3

Combinación	abcd		1's	
(0,2) (0,4)	00-0 0-00		0	
(2,6) (4,6)	0-10 01-0		1	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

NOTA IMPORTANTE: 9 es el único minitérmino que no se combina con otra cosa, y queda sin marca *, así que se toma como un implicante primo que va directo a la tabla 5.

4) Generamos una cuarta tabla. **Objetivo:** Realizar la segunda combinación con las combinaciones de la tabla anterior.

Buscamos combinar a las combinaciones:

Combinación	abcd
(0,2)	00-0
(4,6)	01-0

COMBINACIÓN: 0—0

Tabla 3

Combinación	abcd		1's	
(0,2) (0,4)	00-0 0-00		0	
(2,6) (4,6)	0-10 01-0		1	

Tabla 4

Combinación	abcd		1's	
(0,2,4,6)	0—0		0	

Marcamos con un * para las combinaciones (0,2), (0,4), (2,6) y (4,6), porque al combinarlos dan lo mismo: (0,2,4,6)

5) Generamos la tabla 5. **Objetivo:** Colocar a los <u>implicantes primos</u> (combinaciones que ya no pueden ser combinadas con otras; se denotan por no tener una marca * en las tablas 3 y 4), en una sola tabla para obtener a los productos.

Tabla 5. Implicantes primos

Combinación	abcd		1's		0	2	4	5	6	8	Producto
(0,2,4,6)	0—0		0		X	X	X		X		a'd'
9	1001		0		X					X	ab'c'd

NOTA: Si en la representación binaria (abcd) tenemos para el producto:
- no hay variable representativa
0 la variable está negada
1 la variable no está negada

FINALMENTE OBTENEMOS LA MINIMIZACIÓN DADA POR LA SUMA DE LOS PRODUCTOS, resultado que podemos representar como:

$$f(a,b,c,d) = \sum\nolimits_m (0,2,4,6,9) = a'd' + ab'c'd$$

O simplemente:

$$f(a,b,c,d) = a'd' + ab'c'd$$

EJEMPLO 6

$$f(a,b,c,d) = \sum_m (1,2,5,8,9,10,13)$$

Solución:

1) Generamos la primera tabla. **Objetivo:** Obtener las formas binarias de los minitérminos y la cantidad de 1's en ellos.

Tabla 1

Minitérminos	1's	abcd	
1	1	0001	
2	1	0010	
5	2	0101	
8	1	1000	
9	2	1001	
10	2	1010	
13	3	1101	

- **Columna minitérminos:** llenamos conforme a los minitérminos que nos muestra la función en este caso: $1, 2, 5, 8, 9, 10$ y 13
- **Columna 1´s:** es el número de **1´s** que contiene el número binario de ese minitérmino
- **abcd:** el número binario que representa ese minitérmino

2) Generamos la segunda tabla. **Objetivo:** Reagrupar a los minitérminos con base en la cantidad de 1's en su representación binaria.

PASO 2.1

Tabla 1

Minitérminos	1's	abcd	
1	1	0001	*
2	1	0010	
5	2	0101	
8	1	1000	
9	2	1001	
10	2	1010	
13	3	1101	

Tabla 2

Minitérminos	abcd		1's	
1	0001		1	

Marcamos con un * en la Tabla 1, para indicar que hemos agregado los minitérminos en la Tabla 2.

PASO 2.2

Tabla 1

Minitérminos	1's	abcd	
1	1	0001	*
2	1	0010	*
5	2	0101	
8	1	1000	
9	2	1001	
10	2	1010	
13	3	1101	

Tabla 2

Minitérminos	abcd		1's	
1 2	0001 0010		1	

PASO 2.3

Tabla 1

Minitérminos	1's	abcd	
1	1	0001	*
2	1	0010	*
5	2	0101	*
8	1	1000	
9	2	1001	
10	2	1010	
13	3	1101	

Tabla 2

Minitérminos	abcd		1's	
1 2	0001 0010		1	
5	0101		2	

PASO 2.4

Tabla 1

Minitérminos	1's	abcd	
1	1	0001	*
2	1	0010	*
5	2	0101	*
8	1	1000	*
9	2	1001	
10	2	1010	
13	3	1101	

Tabla 2

Minitérminos	abcd		1's	
1	0001		1	
2	0010			
8	1000			
5	0101		2	

PASO 2.5

Tabla 1

Minitérminos	1's	abcd	
1	1	0001	*
2	1	0010	*
5	2	0101	*
8	1	1000	*
9	2	1001	*
10	2	1010	
13	3	1101	

Tabla 2

Minitérminos	abcd		1's	
1	0001		1	
2	0010			
8	1000			
5	0101		2	
9	1001			

PASO 2.6

Tabla 1

Minitérminos	1's	abcd	
1	1	0001	*
2	1	0010	*
5	2	0101	*
8	1	1000	*
9	2	1001	*
10	2	1010	*
13	3	1101	

Tabla 2

Minitérminos	abcd		1's	
1	0001		1	
2	0010			
8	1000			
5	0101		2	
9	1001			
10	1010			

PASO 2.7

Tabla 1

Minitérminos	1's	abcd	
1	1	0001	*
2	1	0010	*
5	2	0101	*
8	1	1000	*
9	2	1001	*
10	2	1010	*
13	3	1101	*

Tabla 2

Minitérminos	abcd		1's	
1 2 8	0001 0010 1000		1	
5 9 10	0101 1001 1010		2	
13	1101		3	

3) Generamos una tercera tabla. **Objetivo:** Realizar la primera combinación con minitérminos.
**Pasamos a la parte de realización de combinaciones con los minitérminos.

COMBINACIONES:

1. Buscamos en la representación binaria de los minitérminos, los que difieren en solo una variable (la que sea) para poder realizar una combinación con apoyo en la tabla 2.
2. Empezamos con el primer caso en la cantidad de 1's con la siguiente cantidad. En este caso los que tienen cero 1's con los que tiene un solo 1.
3. Al combinar a los minitérminos, si tenemos:
   - 0 y 0 será 0
   - 1 y 1 será 1
   - 0 y 1 será -

Ejemplo de una combinación de minitérminos:

Buscamos combinar al minitérmino 0 y 2.

Minitérmino	abcd
1	0001
5	0101

COMBINACIÓN: 0-01

PASO 3.1

Tabla 2

Minitérminos	abcd		1's	
1	0001	*	1	
2	0010			
8	1000			
5	0101	*	2	
9	1001			
10	1010			
13	1101		3	

Tabla 3

Combinación	abcd		1's	
(1,5)	0-01		1	

Marcamos con un * en la Tabla 2, para indicar que hemos agregado las combinaciones de minitérminos posibles en la Tabla 3.

PASO 3.2

Tabla 2

Minitérminos	abcd		1's	
1	0001	*	1	
2	0010	*		
8	1000			
5	0101	*	2	
9	1001			
10	1010	*		
13	1101		3	

Tabla 3

Combinación	abcd		1's	
(1,5)	0-01		1	
(2,10)	-010			

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

PASO 3.3

Tabla 2

Minitérminos	abcd		1's	
1	0001	*	1	
2	0010	*		
8	1000	*		
5	0101	*	2	
9	1001	*		
10	1010			
13	1101		3	

Tabla 3

Combinación	abcd		1's	
(1,5)	0-01		1	
(2,10)	-010			
(8,9)	100-			

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

PASO 3.4

Tabla 2

Minitérminos	abcd		1's	
1	0001	*	1	
2	0010	*		
8	1000	*		
5	0101	*	2	
9	1001	*		
10	1010	*		
13	1101		3	

Tabla 3

Combinación	abcd		1's	
(1,5)	0-01		1	
(2,10)	-010			
(8,9)	100-			
(8,10)	10-0			

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

PASO 3.5

Tabla 2

Minitérminos	abcd		1's	
1	0001	*	1	
2	0010	*		
8	1000	*		
5	0101	*	2	
9	1001	*		
10	1010	*		
13	1101	*	3	

Tabla 3

Combinación	abcd		1's	
(1,5)	0-01		1	
(2,10)	-010			
(8,9)	100-			
(8,10)	10-0			
(5,13)	-101		2	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

PASO 3.6

Tabla 2

Minitérminos	abcd		1's	
1	0001	*	1	
2	0010	*		
8	1000	*		
5	0101	*	2	
9	1001	*		
10	1010	*		
13	1101	*	3	

Tabla 3

Combinación	abcd		1's	
(1,5)	0-01		1	
(2,10)	-010			
(8,9)	100-			
(8,10)	10-0			
(5,13)	-101		2	
(9,13)	1-01			

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

4) Generamos una cuarta tabla. **Objetivo:** Realizar la segunda combinación con las combinaciones de la tabla anterior.

Buscamos combinar a las combinaciones:

Combinación	abcd
(1,5)	0-01
(9,13)	1-01

COMBINACIÓN: —-01

Tabla 3

Combinación	abcd		1's	
(1,5) (2,10) (8,9) (8,10)	0-01 -010 100- 10-0	*	1	
(5,13) (9,13)	-101 1-01	 *	2	

Tabla 4

Combinación	abcd		1's	
(1,5,9,13)	--01		1	

5) Generamos la tabla 5. **Objetivo:** Colocar a los <u>implicantes primos</u> (combinaciones que ya no pueden ser combinadas con otras; se denotan por no tener una marca * en las tablas 3 y 4), en una sola tabla para obtener a los productos.

Tabla 5. Implicantes primos

Combinación	abcd		1's		1	2	5	8	9	10	13	Producto
(1,5,9,13)	--01		1		X	X			X		X	c'd
(2,10)	-010		1			X				X		b'cd'
(8,9)	100-		1					X	X			ab'c'
(8,10)	10-0		1					X		X		ab'd'
(5,13)	-101		2				X				X	bc'd

*(8,10) ya están considerados en otras combinaciones, por lo tanto no es necesaria.

NOTA: Si en la representación binaria (abcd) tenemos para el producto:
- no hay variable representativa
0 la variable está negada
1 la variable no está negada

FINALMENTE OBTENEMOS LA MINIMIZACIÓN DADA POR LA SUMA DE LOS PRODUCTOS, resultado que podemos representar como:

$$f(a,b,c,d) = \sum_m (1,2,5,8,9,10,13) = c'd + b'cd' + ab'c' + bc'd$$

O simplemente:

$$f(a,b,c,d) = c'd + b'cd' + ab'c' + bc'd$$

EJEMPLO 7

$$f(a,b,c,d) = \sum\nolimits_m (0,2,3,7,9)$$

Solución:

1) Generamos la primera tabla. **Objetivo:** Obtener las formas binarias de los minitérminos y la cantidad de 1's en ellos.

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	
2	1	0010	
3	2	0011	
7	3	0111	
9	2	1001	

- **Columna minitérminos:** llenamos conforme a los minitérminos que nos muestra la función en este caso: $0, 2, 3, 7$ y 9
- **Columna 1´s:** es el número de **1´s** que contiene el número binario de ese minitérmino
- **abcd:** el número binario que representa ese minitérmino

2) Generamos la segunda tabla. **Objetivo:** Reagrupar a los minitérminos con base en la cantidad de 1's en su representación binaria.

PASO 2.1

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	
3	2	0011	
7	3	0111	
9	2	1001	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	

Marcamos con un * en la Tabla 1, para indicar que hemos agregado los minitérminos en la Tabla 2.

PASO 2.2

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
3	2	0011	
7	3	0111	
9	2	1001	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2	0010		1	

PASO 2.3

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
3	2	0011	*
7	3	0111	
9	2	1001	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2	0010		1	
3	0011		2	

PASO 2.4

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
3	2	0011	*
7	3	0111	*
9	2	1001	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2	0010		1	
3	0011		2	
7	0111		3	

PASO 2.5

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
3	2	0011	*
7	3	0111	*
9	2	1001	*

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2	0010		1	
3 9	0011 1001		2	
7	0111		3	

3) Generamos una tercera tabla. **Objetivo:** Realizar la primera combinación con minitérminos.
   **Pasamos a la parte de realización de combinaciones con los minitérminos.

   COMBINACIONES:
   1. Buscamos en la representación binaria de los minitérminos, los que difieren en solo una variable (la que sea) para poder realizar una combinación con apoyo en la tabla 2.
   2. Empezamos con el primer caso en la cantidad de 1's con la siguiente cantidad. En este caso los que tienen cero 1's con los que tiene un solo 1.
   3. Al combinar a los minitérminos, si tenemos:
      0 y 0 será 0
      1 y 1 será 1
      0 y 1 será -

Ejemplo de una combinación de minitérminos:

Buscamos combinar al minitérmino 0 y 2.

Minitérmino	abcd
0	0000
2	0010

COMBINACIÓN: 00-0

PASO 3.1

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
2	0010	*	1	
3 9	0011 1001		2	
7	0111		3	

Tabla 3

Combinación	abcd		1's	
(0,2)	00-0		0	

Marcamos con un * en la Tabla 2, para indicar que hemos agregado las combinaciones de minitérminos posibles en la Tabla 3.

PASO 3.2

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
2	0010	*	1	
3 9	0011 1001	*	2	
7	0111		3	

Tabla 3

Combinación	abcd		1's	
(0,2)	00-0		0	
(2,3)	001-		1	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

PASO 3.3

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
2	0010	*	1	
3 9	0011 1001	*	2	
7	0111		3	

Tabla 3

Combinación	abcd		1's	
(0,2)	00-0		0	
(2,3)	001-		1	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

PASO 3.4

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
2	0010	*	1	
3 9	0011 1001	*	2	
7	0111	*	3	

Tabla 3

Combinación	abcd		1's	
(0,2)	00-0		0	
(2,3)	001-		1	
(3,7)	0-11		2	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

NOTA: 9 no se puede combinar con los otros minitérminos, por lo tanto: es un implicante primo, y va directo a la tabla 5.

4) Generamos una cuarta tabla. **Objetivo:** Realizar la segunda combinación con las combinaciones de la tabla anterior.

Tabla 3

Combinación	abcd		1's	
(0,2)	00-0		0	
(2,3)	001-		1	
(3,7)	0-11		2	

Tabla 4

Combinación	abcd		1's	

NO SE PUEDEN HACER COMBINACIONES DE LAS COMBINACIONES QUE TENEMOS.

5) Generamos la tabla 5. **Objetivo:** Colocar a los <u>implicantes primos</u> (combinaciones que ya no pueden ser combinadas con otras; se denotan por no tener una marca * en las tablas 3 y 4), en una sola tabla para obtener a los productos.

Tabla 5. Implicantes primos

Combinación	abcd		1's		0	2	3	7	9	Producto
(0,2)	00-0		0		X	X				a'b'd'
(2,3)	001-		1			X	X			a'b'c
(3,7)	0-11		2				X	X		a'cd
(9)	1001		2						X	ab'c'd

*(2,3) ya están considerados en otras combinaciones, por lo tanto no es necesaria

NOTA: Si en la representación binaria (abcd) tenemos para el producto:
- no hay variable representativa
0 la variable está negada
1 la variable no está negada

FINALMENTE OBTENEMOS LA MINIMIZACIÓN DADA POR LA SUMA DE LOS PRODUCTOS, resultado que podemos representar como:

$$f(a,b,c,d) = \sum_m (0,2,3,7,9) = a'b'd' + a'cd + ab'c'd$$

O simplemente:

$$f(a,b,c,d) = a'b'd' + a'cd + ab'c'd$$

EJEMPLO 8

$$f(a,b,c,d) = \sum\nolimits_m (4,5,7,13)$$

Solución:

1) Generamos la primera tabla. **Objetivo:** Obtener las formas binarias de los minitérminos y la cantidad de 1's en ellos.

Tabla 1

Minitérminos	1's	abcd	
4	1	0100	
5	2	0101	
7	3	0111	
13	3	1101	

- **Columna minitérminos:** llenamos conforme a los minitérminos que nos muestra la función en este caso: $4, 5, 7$ y 13
- **Columna 1´s:** es el número de **1´s** que contiene el número binario de ese minitérmino
- **abcd:** el número binario que representa ese minitérmino

2) Generamos la segunda tabla. **Objetivo:** Reagrupar a los minitérminos con base en la cantidad de 1's en su representación binaria.

PASO 2.1

Tabla 1

Minitérminos	1's	abcd	
4	1	0100	*
5	2	0101	
7	3	0111	
13	3	1101	

Tabla 2

Minitérminos	abcd		1's	
4	0100		1	

Marcamos con un * en la Tabla 1, para indicar que hemos agregado los minitérminos en la Tabla 2.

PASO 2.2

Tabla 1

Minitérminos	1's	abcd	
4	1	0100	*
5	2	0101	*
7	3	0111	
13	3	1101	

Tabla 2

Minitérminos	abcd		1's	
4	0100		1	
5	0101		2	

PASO 2.3

Tabla 1

Minitérminos	1's	abcd	
4	1	0100	*
5	2	0101	*
7	3	0111	*
13	3	1101	

Tabla 2

Minitérminos	abcd		1's	
4	0100		1	
5	0101		2	
7	0111		3	

PASO 2.4

Tabla 1

Minitérminos	1's	abcd	
4	1	0100	*
5	2	0101	*
7	3	0111	*
13	3	1101	*

Tabla 2

Minitérminos	abcd		1's	
4	0100		1	
5	0101		2	
7 13	0111 1101		3	

3) Generamos una tercera tabla. **Objetivo:** Realizar la primera combinación con minitérminos.
**Pasamos a la parte de realización de combinaciones con los minitérminos.

COMBINACIONES:
1. Buscamos en la representación binaria de los minitérminos, los que difieren en solo una variable (la que sea) para poder realizar una combinación con apoyo en la tabla 2.
2. Empezamos con el primer caso en la cantidad de 1's con la siguiente cantidad. En este caso los que tienen cero 1's con los que tiene un solo 1.
3. Al combinar a los minitérminos, si tenemos:
   0 y 0 será 0
   1 y 1 será 1
   0 y 1 será -

Ejemplo de una combinación de minitérminos:
Buscamos combinar al minitérmino 0 y 2.

Minitérmino	abcd
4	0100
5	0101

COMBINACIÓN: 010-

PASO 3.1

Tabla 2

Minitérminos	abcd		1's	
4	0100	*	1	
5	0101	*	2	
7 13	0111 1101		3	

Tabla 3

Combinación	abcd		1's	
(4,5)	010-		1	

Marcamos con un * en la Tabla 2, para indicar que hemos agregado las combinaciones de minitérminos posibles en la Tabla 3.

PASO 3.2

Tabla 2

Minitérminos	abcd		1's	
4	0100	*	1	
5	0101	*	2	
7 13	0111 1101	*	3	

Tabla 3

Combinación	abcd		1's	
(4,5)	010-		1	
(5,7)	01-1		2	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

PASO 3.3

Tabla 2

Minitérminos	abcd		1's	
4	0100	*	1	
5	0101	*	2	
7 13	0111 1101	* *	3	

Tabla 3

Combinación	abcd		1's	
(4,5)	010-		1	
(5,7) (5,13)	01-1 -101		2	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

4) Generamos una cuarta tabla. **Objetivo:** Realizar la segunda combinación con las combinaciones de la tabla anterior.

Tabla 3

Combinación	abcd		1's	
(4,5)	010-		1	
(5,7) (5,13)	01-1 -101		2	

Tabla 4

Combinación	abcd		1's	

NO SE PUEDEN HACER COMBINACIONES DE LAS COMBINACIONES QUE TENEMOS. Vamos directo a la tabla 5.

5) Generamos la tabla 5. **Objetivo:** Colocar a los implicantes primos (combinaciones que ya no pueden ser combinadas con otras; se denotan por no tener una marca * en las tablas 3 y 4), en una sola tabla para obtener a los productos.

Tabla 5. Implicantes primos

Combinación	abcd		1's		4	5	7	13	Producto
(4,5)	010-		1		X	X			ab'c
(5,7)	01-1		2			X	X		a'bd
(5,13)	-101		2			X		X	bc'd

NOTA: Si en la representación binaria (abcd) tenemos para el producto:
- no hay variable representativa
0 la variable está negada
1 la variable no está negada

FINALMENTE OBTENEMOS LA MINIMIZACIÓN DADA POR LA SUMA DE LOS PRODUCTOS, resultado que podemos representar como:

$$f(a,b,c,d) = \sum_m (4,5,7,13) = ab'c + a'bd + bc'd$$

O simplemente:

$$f(a,b,c,d) = ab'c + a'bd + bc'd$$

EJEMPLO 9

$$f(a,b,c,d) = \sum\nolimits_m (0,2,4,5,6)$$

Solución:

1) Generamos la primera tabla. **Objetivo:** Obtener las formas binarias de los minitérminos y la cantidad de 1's en ellos.

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	
2	1	0010	
4	1	0100	
5	2	0101	
6	2	0110	

- **Columna minitérminos:** llenamos conforme a los minitérminos que nos muestra la función en este caso: $0, 2, 4, 5$ y 6
- **Columna 1´s:** es el número de **1´s** que contiene el número binario de ese minitérmino
- **abcd:** el número binario que representa ese minitérmino

2) Generamos la segunda tabla. **Objetivo:** Reagrupar a los minitérminos con base en la cantidad de 1's en su representación binaria.

PASO 2.1

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	
4	1	0100	
5	2	0101	
6	2	0110	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	

Marcamos con un * en la Tabla 1, para indicar que hemos agregado los minitérminos en la Tabla 2.

PASO 2.2

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
4	1	0100	
5	2	0101	
6	2	0110	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2	0010		1	

PASO 2.3

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
4	1	0100	*
5	2	0101	
6	2	0110	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2 4	0010 0100		1	

PASO 2.4

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
4	1	0100	*
5	2	0101	*
6	2	0110	

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2 4	0010 0100		1	
5	0101		2	

PASO 2.5

Tabla 1

Minitérminos	1's	abcd	
0	0	0000	*
2	1	0010	*
4	1	0100	*
5	2	0101	*
6	2	0110	*

Tabla 2

Minitérminos	abcd		1's	
0	0000		0	
2 4	0010 0100		1	
5 6	0101 0110		2	

3) Generamos una tercera tabla. **Objetivo:** Realizar la primera combinación con minitérminos.
   **Pasamos a la parte de realización de combinaciones con los minitérminos.

   COMBINACIONES:
   1. Buscamos en la representación binaria de los minitérminos, los que difieren en solo una variable (la que sea) para poder realizar una combinación con apoyo en la tabla 2.
   2. Empezamos con el primer caso en la cantidad de 1's con la siguiente cantidad. En este caso los que tienen cero 1's con los que tiene un solo 1.
   3. Al combinar a los minitérminos, si tenemos:
      0 y 0 será 0
      1 y 1 será 1
      0 y 1 será -

Ejemplo de una combinación de minitérminos:

Buscamos combinar al minitérmino 0 y 2.

Minitérmino	abcd
0	0000
2	0010

COMBINACIÓN: 00-0

PASO 3.1

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
2 4	0010 0100	*	1	
5 6	0101 0110		2	

Tabla 3

Combinación	abcd		1's	
(0,2)	00-0		0	

Marcamos con un * en la Tabla 2, para indicar que hemos agregado las combinaciones de minitérminos posibles en la Tabla 3.

PASO 3.2

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
2 4	0010 0100	* *	1	
5 6	0101 0110	*	2	

Tabla 3

Combinación	abcd		1's	
(0,2)	00-0		0	
(4,5)	010-		1	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

PASO 3.3

Tabla 2

Minitérminos	abcd		1's	
0	0000	*	0	
2 4	0010 0100	* *	1	
5 6	0101 0110	* *	2	

Tabla 3

Combinación	abcd		1's	
(0,2)	00-0		0	
(4,5) (4,6)	010- 01-0		1	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

4) Generamos una cuarta tabla. **Objetivo:** Realizar la segunda combinación con las combinaciones de la tabla anterior.

Buscamos combinar a las combinaciones:

Combinación	abcd
(0,2)	00-0
(4,6)	01-0

COMBINACIÓN: 0—0

Tabla 3

Combinación	abcd		1's	
(0,2)	00-0	*	0	
(4,5) (4,6)	010- 01-0	 *	1	

Tabla 4

Combinación	abcd		1's	
(0,2,4,6)	0--0		0	

5) Generamos la tabla 5. **Objetivo:** Colocar a los <u>implicantes primos</u> (combinaciones que ya no pueden ser combinadas con otras; se denotan por no tener una marca * en las tablas 3 y 4), en una sola tabla para obtener a los productos.

Tabla 5. Implicantes primos

Combinación	abcd		1's		0	2	4	5	6	Producto
(0,2,4,6)	0--0		0		X	X	X		X	a'd'
(4,5)	010-		1				X	X		a'bc'

NOTA: Si en la representación binaria (abcd) tenemos para el producto:
- no hay variable representativa
0 la variable está negada
1 la variable no está negada

FINALMENTE OBTENEMOS LA MINIMIZACIÓN DADA POR LA SUMA DE LOS PRODUCTOS, resultado que podemos representar como:

$$f(a,b,c,d) = \sum_m (0,2,3,7,9) = a'd' + abc'$$

O simplemente:

$$f(a,b,c,d) = a'd' + abc'$$

EJEMPLO 10

$$f(a,b,c,d) = \sum\nolimits_m (1,3,5,7,9)$$

Solución:

1) Generamos la primera tabla. **Objetivo:** Obtener las formas binarias de los minitérminos y la cantidad de 1's en ellos.

Tabla 1

Minitérminos	1's	abcd	
1	1	0000	
3	2	0011	
5	2	0101	
7	3	0111	
9	2	1001	

- **Columna minitérminos:** llenamos conforme a los minitérminos que nos muestra la función en este caso: $1, 3, 5, 7\ y\ 9$
- **Columna 1´s:** es el número de **1´s** que contiene el número binario de ese minitérmino
- **abcd:** el número binario que representa ese minitérmino

2) Generamos la segunda tabla. **Objetivo:** Reagrupar a los minitérminos con base en la cantidad de 1's en su representación binaria.

PASO 2.1

Tabla 1

Minitérminos	1's	abcd	
1	1	0000	*
3	2	0011	
5	2	0101	
7	3	0111	
9	2	1001	

Tabla 2

Minitérminos	abcd		1's	
1	0001		1	

Marcamos con un * en la Tabla 1, para indicar que hemos agregado los minitérminos en la Tabla 2.

PASO 2.2

Tabla 1

Minitérminos	1's	abcd	
1	1	0000	*
3	2	0011	*
5	2	0101	
7	3	0111	
9	2	1001	

Tabla 2

Minitérminos	abcd		1's	
1	0001		1	
3	0011		2	

PASO 2.3

Tabla 1

Minitérminos	1's	abcd	
1	1	0000	*
3	2	0011	*
5	2	0101	*
7	3	0111	
9	2	1001	

Tabla 2

Minitérminos	abcd		1's	
1	0001		1	
3 5	0011 0101		2	

PASO 2.4

Tabla 1

Minitérminos	1's	abcd	
1	1	0000	*
3	2	0011	*
5	2	0101	*
7	3	0111	*
9	2	1001	

Tabla 2

Minitérminos	abcd		1's	
1	0001		1	
3 5	0011 0101		2	
7	0111		3	

PASO 2.5

Tabla 1

Minitérminos	1's	abcd	
1	1	0000	*
3	2	0011	*
5	2	0101	*
7	3	0111	*
9	2	1001	*

Tabla 2

Minitérminos	abcd		1's	
1	0001		1	
3 5 9	0011 0101 1001		2	
7	0111		3	

3) Generamos una tercera tabla. **Objetivo:** Realizar la primera combinación con minitérminos.
**Pasamos a la parte de realización de combinaciones con los minitérminos.

COMBINACIONES:

1. Buscamos en la representación binaria de los minitérminos, los que difieren en solo una variable (la que sea) para poder realizar una combinación con apoyo en la tabla 2.
2. Empezamos con el primer caso en la cantidad de 1's con la siguiente cantidad. En este caso los que tienen cero 1's con los que tiene un solo 1.
3. Al combinar a los minitérminos, si tenemos:
   0 y 0 será 0
   1 y 1 será 1
   0 y 1 será -

Ejemplo de una combinación de minitérminos:

Buscamos combinar al minitérmino 0 y 2.

Minitérmino	abcd
0	0000
2	0010

COMBINACIÓN: 00-0

PASO 3.1

Tabla 2

Minitérminos	abcd		1's	
1	0001	*	1	
3 5 9	0011 0101 1001	*	2	
7	0111		3	

Tabla 3

Combinación	abcd		1's	
(1,3)	00-1		1	

Marcamos con un * en la Tabla 2, para indicar que hemos agregado las combinaciones de minitérminos posibles en la Tabla 3.

PASO 3.2

Tabla 2

Minitérminos	abcd		1's	
1	0001	*	1	
3 5 9	0011 0101 1001	* *	2	
7	0111		3	

Tabla 3

Combinación	abcd		1's	
(1,3) (1,5)	00-1 0-01		1	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

PASO 3.3

Tabla 2

Minitérminos	abcd		1's	
1	0001	*	1	
3 5 9	0011 0101 1001	* * *	2	
7	0111		3	

Tabla 3

Combinación	abcd		1's	
(1,3) (1,5) (1,9)	00-1 0-01 -001		1	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

PASO 3.3

Tabla 2

Minitérminos	abcd		1's	
1	0001	*	1	
3 5 9	0011 0101 1001	* * *	2	
7	0111	*	3	

Tabla 3

Combinación	abcd		1's	
(1,3) (1,5) (1,9)	00-1 0-01 -001		1	
(3,7)	0-11		2	

Marcamos con un * en la Tabla 2, para el/los minitérmino(s) que se están combinando.

4) Generamos una cuarta tabla. **Objetivo:** Realizar la segunda combinación con las combinaciones de la tabla anterior.

Buscamos combinar a las combinaciones:

Combinación	abcd
(1,5)	0-01
(3,7)	0-11

COMBINACIÓN: 0—1

Tabla 3

Combinación	abcd		1's	
(1,3) (1,5) (1,9)	00-1 0-01 -001	* *	1	
(3,7)	0-11	*	2	

Tabla 4

Combinación	abcd		1's	
(1,3,5,7)	0--1		1	

Marcamos con un * a las combinaciones que tienen los miniterminos de la combinación (1,3,5,7)

5) Generamos la tabla 5. **Objetivo:** Colocar a los <u>implicantes primos</u> (combinaciones que ya no pueden ser combinadas con otras; se denotan por no tener una marca * en las tablas 3 y 4), en una sola tabla para obtener a los productos.

Tabla 5. Implicantes primos

Combinación	abcd		1's		1	3	5	7	9	Producto
(1,3,5,7)	0--1		1		X	X	X		X	a'd
(1,9)	-001		2			X		X		b'c'd

NOTA: Si en la representación binaria (abcd) tenemos para el producto:
- no hay variable representativa
0 la variable está negada
1 la variable no está negada

FINALMENTE OBTENEMOS LA MINIMIZACIÓN DADA POR LA SUMA DE LOS PRODUCTOS, resultado que podemos representar como:

$$f(a,b,c,d) = \sum\nolimits_m (1,3,5,7,9) = a'd + b'c'd$$

O simplemente:

$$f(a,b,c,d) = a'd + b'c'd$$

CUESTIONARIO

1. **¿Cuáles son los dos métodos principales de minimización utilizados en la síntesis de circuitos lógicos?**

   a) Mapa de Karnaugh y el algoritmo Quine-McCluskey (método tabula).

   b) Método Simplex y Método algebraico

   c) Método gráfico de mapas de Karnaugh y Algebra de Boole

2. **¿Cómo difieren en su aplicabilidad para funciones de más de 4 variables el mapa de Karnaugh y el método tabula (Quine-McCluskey) en la síntesis de circuitos lógicos?**

   a) Mientras que el mapa de Karnaugh se vuelve complicado de analizar visualmente para funciones con más de 4 variables, el método tabula (Quine-McCluskey) es más adecuado y no presenta dificultades significativas al manejar funciones lógicas con un mayor número de variables.
   b) Mientras más complejo el mapa de Karnaugh, más complejo se vuelve el método de Quine McCluskey.
   c) No difieren en ningún momento

3. **¿Cuál es el Algoritmo Quine-McCluskey?**

   a) El algoritmo Quine-McCluskey es un método de simplificación de funciones booleanas desarrollado por Willar Van Orman Quine y Edward J. McCluskey
   b) Es un algoritmo de un método gráfico de simplificación de funciones logarítmicas desarrollado por Edward J. McCluskey y Bernhard Riemann
   c) Es un método de simplificación de funciones desarrollado por Edward J. McCluskey y Grigori Perelmán

4. **¿Cuáles son pasos principales en la simplificación de funciones booleanas? Elige 2**

   1. Involucra encontrar todos los implicantes primos de la función.
   2. Consiste en identificar los implicantes primos esenciales, que son necesarios y suficientes para generar la función simplificada.
   3. Consiste en encontrar todos los implicantes irracionales que existen en la función
   4. Encontrar los elementos fraccionarios dentro de la función simplificada

5. **¿Qué es la representación cúbica de las funciones de Boole?**

a) La representación cubica de las funciones de Boole es una técnica gráfica que utiliza cubos para visualizar las combinaciones posibles de variables booleanas. En esta representación, una variable se muestra como un punto en un segmento, dos variables en un cuadrado, y tres variables en un cubo.
b) Es una técnica tabular que utiliza cubos para visualizar las combinaciones posibles de variables.
c) Representación de una sola combinación de una variable booleana

6. **¿Qué son los implicantes primos en el contexto del método de Quine-McCluskey?**

a) Los implicantes primos son cubos que no están completamente contenidos en otros cubos más grandes de una función booleana. Estos cubos se utilizan para simplificar la función.
b) Los implicantes primos son cubos que se encuentran contenidos en cubos más pequeños de una función boolena.
c) Los implicantes primos son cubos que se encuentran contenidos en cubos más pequeños de una función boolena. Estos cubos se utilizan para simplificar la función

7. **¿Cómo se simplifican las funciones booleanas utilizando el método de Quine-McCluskey?**

a) El método de Quine-McCluskey simplifica funciones booleanas comparando mintérminos y combinándolos en cubos de orden superior. Se buscan los implicantes primos esenciales que cubren los mintérminos de la función, y luego se buscan implicantes primos secundarios de menor costo. La función simplificada se obtiene combinando estos implicantes primos.
b) Se buscan implicantes compuestos al azar y luego implicantes primos esenciales, al combinarlos se obtiene la función simplificada
c) Selecciona un grupo de implicantes primos de menor valor y otro grupo de estos mismos, al final se combinan y la función queda simplificada

8. **¿Qué papel juegan los cubos en la representación cúbica de funciones booleanas?**

1. Los cubos se utilizan en la representación cúbica para visualizar las combinaciones posibles de variables booleanas. Cada cubo representa una combinación de variables y se utiliza para simplificar funciones booleanas mediante el método de Quine-McCluskey.
2. Cada puede representar distintas combinaciones de variables y se utiliza para simplificar funciones booleanas

3. Son una representación cubica que visualiza una sola combinación de variables booleanas. Cada cubo es igual a diferentes combinaciones de variables y se utiliza en el método de Quine-McCluskey.

9. ¿Cuál es la ventaja de simplificar funciones booleanas utilizando el método de Quine- McCluskey?

a) Obtener expresiones más compactas y eficientes, lo que facilita su implementación en circuitos electrónicos y reduce la complejidad. Esto mejora el rendimiento y reduce el costo de los circuitos lógicos.
b) Obtener expresiones más complejas pero eficientes al implementarlas en circuitos eléctricos.
c) Un mejor rendimiento en los circuitos eléctricos, pero aumentando su costo

10. ¿Qué implicantes se consideran primos esenciales en el algoritmo de Quine-McCluskey?

a) Los implicantes primos esenciales son aquellos que no pueden ser reducidos más y que tienen solo un tache en la columna de números binarios.
b) Son aquellos que si se pueden reducir y que tienen múltiples taches en la columna de números binarios.
c) Aquellos que no pueden ser reducidos más y que tienen múltiples taches en la columna de números binarios

11. ¿Cuál es el propósito de la minimización de funciones booleanas en el diseño de circuitos lógicos?

a) La minimización de funciones boolenas es esencial en el diseño de circuitos lógicos ya que afecta la complejidad del sistema, su costo y su implementación. El objetivo es representar una función booleana como la suma del menor número de términos, lo que permite simplificar el diseño de circuitos lógicos.
b) Su objetivo es representar una función booleana como la resta del número mayor de términos, lo que nos daría un diseño más complejo, pero más útil.
c) No tienen ningún objetivo, simplemente es tener el circuito de manera visual.

12. ¿Qué ventajas se mencionan en la versión decimal del algoritmo Quine-McCluskey con la modificación propuesta?

a) Si importa con la necesidad de utilizar números adicionales en cada paso del algoritmo
b) La ventaja es que permite tratar aritméticamente los términos "No-Importa" sin la necesidad de utilizar números adicionales en cada paso del algoritmo.
c) No tiene ninguna ventaja la versión decimal del algoritmo Quine McCluskey

13. ¿Cómo se lleva a cabo la simplificación de funciones booleanas en la versión decimal del algoritmo Quine-McCluskey?

a) Lo términos se simplifican comparando sus valores decimales con los enteros y si la diferencia entre estos dos términos es múltiplo de 5, se genera un termino reducido
b) Los términos se simplifican comparando sus valores decimales, y si la diferencia entre dos términos es un múltiplo de 3, pueden generar un término reducido.
c) Tomamos nuestro primer número decimal y si es múltiplo de 6, podemos generar su término reducido

13. ¿Cómo difieren en su aplicabilidad para funciones de más de 4 variables el mapa de Karnaugh y el método tabula (Quine-McCluskey) en la síntesis de circuitos lógicos?

1. El método Quine Mccluskey presenta ciertas dificultades al manejar funciones lógicas con un menor número de variables.
2. Mientras que el mapa de Karnaugh se vuelve complicado de analizar visualmente para funciones con más de 4 variables, el método tabula (Quine-McCluskey) es más adecuado y no presenta dificultades significativas al manejar funciones lógicas con un mayor número de variables.
3. No difiere en ningún caso

14. ¿Qué es la representación cúbica de las funciones de Boole?

1. La representación cúbica de las funciones de Boole es una técnica gráfica que utiliza cubos para visualizar las combinaciones posibles de variables booleanas.
2. Es una representación tabular de las funciones de Boole que visualiza las combinaciones posibles de diferentes variables
3. Es una técnica tabular que utiliza cubos para visualizar una sola variable booleana

15. ¿Qué son los implicantes primos en el contexto del método de Quine-McCluskey?

1. Los implicantes primos son cubos que no están completamente contenidos en otros cubos más grandes de una función booleana. Estos cubos se utilizan para simplificar la función.
2. Los implicantes son cubos contenidos en cubos más pequeños de una función booleana. Se utilizan para simplificar la función
3. Cubos contenidos en cubos más grandes de una función y estos cubos hacen más compleja nuestra función, pero con más opciones para trabajarla.

16. **¿Cómo se simplifican las funciones booleanas utilizando el método de Quine-McCluskey?**
    1. comparando minitérminos y combinándolos en cubos de orden superior. Se buscan los implicantes primos esenciales que cubren los mintérminos de la función, y luego se buscan implicantes primos secundarios de menor costo. La función simplificada se obtiene combinando estos implicantes primos.
    2. No podemos simplificarlas, simplemente tenerlas de manera visual.
    3. Combinando los implicantes primos.

17. **¿Qué papel juegan los cubos en la representación cúbica de funciones booleanas? Respuesta:**
    1. Los cubos se utilizan en la representación cúbica para visualizar las combinaciones posibles de variables booleanas. Cada cubo representa una combinación de variables y se utiliza para simplificar funciones booleanas mediante el método de Quine-McCluskey.
    2. Cada cubo representa distintas combinaciones de variables y nos muestra las funciones de manera más compleja.
    3. Los cubos nos muestran funciones más complejas, pero más útiles para trabajarlas.

18. **¿Cuál es la ventaja de simplificar funciones booleanas utilizando el método de Quine- McCluskey?**
    1. La ventaja de simplificar funciones booleanas con el método de Quine-McCluskey es obtener expresiones más compactas y eficientes, lo que facilita su implementación en circuitos electrónicos y reduce la complejidad. Esto mejora el rendimiento y reduce el costo de los circuitos lógicos.
    2. No hay ninguna ventaja, al contrario, es más complejo utilizar el método de Quine- McCluskey.
    3. La ventaja de tener una función booleana más compleja pero más sencilla su implementación en circuitos electrónicos

19. **¿Qué implicantes se consideran primos esenciales en el algoritmo de Quine-McCluskey?**
    1. Son aquellos que no pueden ser reducidos más y que tienen solo un tache en la columna de números binarios.
    2. Son aquellos que podemos reducir más y tienen múltiples taches en la columna de números binarios
    3. Son aquellos que podemos reducir más y que tienen un solo tache en su columna de números binarios.

20. **¿Cuál es el propósito de la minimización de funciones booleanas en el diseño de circuitos lógicos?**
    1. La minimización de funciones booleanas es esencial en el diseño de circuitos lógicos ya que afecta la complejidad del sistema, su costo y su implementación. El objetivo es representar una función booleana como la suma del menor número de términos, lo que permite simplificar el diseño de circuitos lógicos.
    2. Su objetivo es representar una función booleana como el producto del mayor número de términos, lo que permite simplificar el diseño de los circuitos lógicos.
    3. No existe ningún propósito

21. **¿Quiénes fueron los desarrolladores del Algoritmo de Quine-McCluskey?**
    Respuesta:
    a) El Algoritmo de Quine-McCluskey fue desarrollado por Willard Van Orman Quine y Edward J. McCluskey. (Correcta)
    b) El Algoritmo de Quine-McCluskey fue desarrollado por George Boole y John McCluskey.
    c) El Algoritmo de Quine-McCluskey fue desarrollado por Alan Turing y Charles Babbage.

22. **¿Cuál es el propósito principal del Algoritmo de Quine-McCluskey en la simplificación de funciones booleanas?**
    Respuesta:
    a) El propósito principal del Algoritmo de Quine-McCluskey es simplificar funciones booleanas, reduciendo la complejidad y el número de términos en sus expresiones. (Correcta)
    b) El propósito principal del Algoritmo de Quine-McCluskey es aumentar la complejidad de las funciones booleanas.
    c) El propósito principal del Algoritmo de Quine-McCluskey es generar funciones booleanas más largas y complejas.

23. **¿Cuál es la entrada típica para el Algoritmo de Quine-McCluskey?**
    Respuesta:

    a) La entrada típica para el Algoritmo de Quine-McCluskey son los mintérminos de una función booleana que se desea simplificar. (Correcta)
    b) La entrada típica para el Algoritmo de Quine-McCluskey son los números primos de una función booleana.

c) La entrada típica para el Algoritmo de Quine-McCluskey son los números enteros de una función booleana.

24. **¿Qué es un mintérmino en el contexto del Algoritmo de Quine-McCluskey?**
Respuesta:
a) Un mintérmino es una expresión booleana que consiste en la productoria (AND) de todas las variables de entrada en su forma original o complementada. (Correcta)
b) Un mintérmino es una expresión booleana que consiste en la suma (OR) de todas las variables de entrada en su forma original o complementada.
c) Un mintérmino es una expresión booleana que consiste en la resta (RESTA) de todas las variables de entrada en su forma original o complementada.

25. **¿Cuál es el primer paso del Algoritmo de Quine-McCluskey?**
Respuesta:

a) El primer paso del Algoritmo de Quine-McCluskey implica encontrar todos los implicantes primos de la función booleana. (Correcta)
b) El primer paso del Algoritmo de Quine-McCluskey consiste en simplificar la función booleana de inmediato.
c) El primer paso del Algoritmo de Quine-McCluskey es buscar los implicantes primos esenciales.

26. **¿Qué son los implicantes primos en el contexto del Algoritmo de Quine-McCluskey?**
Respuesta:
a) Los implicantes primos son términos irreducibles que cubren una o varias combinaciones de mintérminos en la función booleana. (Correcta)
b) Los implicantes primos son términos que no tienen relación con los mintérminos en la función booleana.
c) Los implicantes primos son términos complementarios a los mintérminos en la función booleana.

27. **¿Cuál es el segundo paso del Algoritmo de Quine-McCluskey?**
    Respuesta:
    a) El segundo paso del Algoritmo de Quine-McCluskey consiste en identificar los implicantes primos esenciales, que son necesarios y suficientes para generar la función simplificada. (Correcta)
    b) El segundo paso del Algoritmo de Quine-McCluskey es combinar todos los términos de la función en una única expresión.
    c) El segundo paso del Algoritmo de Quine-McCluskey es buscar los mintérminos que no son necesarios para la función simplificada.

28. **¿Qué se hace después de encontrar los implicantes primos esenciales en el Algoritmo de Quine-McCluskey?**
    Respuesta:
    a) Después de encontrar los implicantes primos esenciales, se combinan para obtener la expresión simplificada de la función booleana. (Correcta)
    b) Después de encontrar los implicantes primos esenciales, se eliminan para reducir la complejidad de la función booleana.
    c) Después de encontrar los implicantes primos esenciales, se utilizan como entrada en otro algoritmo para simplificar aún más la función booleana.

29. **¿Cuál es el beneficio de utilizar el Algoritmo de Quine-McCluskey en lugar de otras técnicas de simplificación de funciones booleanas?**
    Respuesta:
    a) El beneficio de utilizar el Algoritmo de Quine-McCluskey en lugar de otras técnicas de simplificación de funciones booleanas es que produce simplificaciones óptimas, lo que significa que proporciona la expresión más reducida y eficiente de una función booleana. (Correcta)
    b) El beneficio de utilizar el Algoritmo de Quine-McCluskey es que es más rápido que otras técnicas, aunque no siempre produce la simplificación más eficiente.
    c) El beneficio de utilizar el Algoritmo de Quine-McCluskey es que es fácil de aprender y aplicar, lo que lo hace adecuado para principiantes en electrónica digital.

30. **¿Qué ocurre si no se encuentran implicantes primos esenciales en el Algoritmo de Quine-McCluskey?**
Respuesta:
   a) Si no se encuentran implicantes primos esenciales, se deben buscar implicantes primos secundarios para obtener una simplificación adecuada. (Correcta)
   b) Si no se encuentran implicantes primos esenciales, la función no se puede simplificar utilizando el Algoritmo de Quine-McCluskey.
   c) Si no se encuentran implicantes primos esenciales, se debe agregar más variables a la función para permitir la simplificación.

31. **¿Cuál es la complejidad computacional del Algoritmo de Quine-McCluskey en términos de tiempo?**
Respuesta:
   a) La complejidad computacional del Algoritmo de Quine-McCluskey es exponencial en el peor de los casos, lo que significa que puede ser lento para funciones con un gran número de variables. (Correcta)
   b) La complejidad computacional del Algoritmo de Quine-McCluskey es lineal en términos de tiempo, lo que lo hace rápido y eficiente.
   c) La complejidad computacional del Algoritmo de Quine-McCluskey es constante, lo que significa que su rendimiento es constante independientemente del tamaño de la función.

32. **¿Qué modificaciones se han propuesto para mejorar el Algoritmo de Quine-McCluskey?**
Respuesta:
   a) Se han propuesto modificaciones, como la versión decimal del algoritmo, que simplifican el tratamiento de los términos "No-Importa". (Correcta)
   b) Se han propuesto modificaciones para aumentar la complejidad del Algoritmo de Quine-McCluskey y hacerlo más eficiente.
   c) Se han propuesto modificaciones para eliminar completamente los términos "No-Importa" del algoritmo.

33. **¿Qué ventajas se mencionan en la versión decimal del Algoritmo de Quine-McCluskey con la modificación propuesta?**
    Respuesta:
    a) La ventaja de la versión decimal del Algoritmo de Quine-McCluskey con la modificación propuesta es que permite tratar aritméticamente los términos "No-Importa" sin la necesidad de utilizar números adicionales en cada paso del algoritmo. (Correcta)
    b) La ventaja de la versión decimal del Algoritmo de Quine-McCluskey con la modificación propuesta es que simplifica la identificación de los implicantes primos esenciales.
    c) La ventaja de la versión decimal del Algoritmo de Quine-McCluskey con la modificación propuesta es que elimina por completo los términos "No-Importa".

34. **¿Cómo se lleva a cabo la simplificación de funciones booleanas en la versión decimal del Algoritmo de Quine-McCluskey?**
    Respuesta:
    a) En la versión decimal del Algoritmo de Quine-McCluskey, los términos se simplifican comparando sus valores decimales, y si la diferencia entre dos términos es un múltiplo de 3, pueden generar un término reducido. (Correcta)
    b) En la versión decimal del Algoritmo de Quine-McCluskey, los términos se simplifican eliminando los valores decimales y reduciendo la función directamente.
    c) En la versión decimal del Algoritmo de Quine-McCluskey, los términos se simplifican mediante una representación en base 10 de los valores booleanos, lo que facilita la simplificación.

35. **¿Cuál es la principal desventaja del Algoritmo de Quine-McCluskey?**
    Respuesta:
    a) La principal desventaja del Algoritmo de Quine-McCluskey es su complejidad exponencial en el peor de los casos, lo que lo hace ineficiente para funciones con muchas variables. (Correcta)
    b) La principal desventaja del Algoritmo de Quine-McCluskey es su falta de precisión en la simplificación de funciones booleanas.
    c) La principal desventaja del Algoritmo de Quine-McCluskey es su incompatibilidad con la representación decimal de funciones booleanas.

36. **¿En qué aplicaciones se utiliza comúnmente el Algoritmo de Quine-McCluskey?**
Respuesta:

a) El Algoritmo de Quine-McCluskey se utiliza en el diseño de circuitos lógicos, en la simplificación de funciones booleanas en electrónica digital y en la optimización de expresiones lógicas. (Correcta)
b) El Algoritmo de Quine-McCluskey se utiliza en aplicaciones médicas para el análisis de datos clínicos.
c) El Algoritmo de Quine-McCluskey se utiliza en la industria de la construcción para el cálculo de estructuras.

37. **¿Cuál es el objetivo final del Algoritmo de Quine-McCluskey?**
Respuesta:

a) El objetivo final del Algoritmo de Quine-McCluskey es encontrar una expresión simplificada que represente de manera eficiente una función booleana dada. (Correcta)
b) El objetivo final del Algoritmo de Quine-McCluskey es encontrar una expresión que aumente la complejidad de una función booleana.
c) El objetivo final del Algoritmo de Quine-McCluskey es encontrar todas las posibles combinaciones de variables en una función booleana.

38. **¿Qué es el principio fundamental detrás del Algoritmo de Quine-McCluskey?**
Respuesta:

a) El principio fundamental detrás del Algoritmo de Quine-McCluskey es encontrar cubos que cubran la máxima cantidad de mintérminos en la función y combinarlos para obtener la simplificación óptima. (Correcta)
b) El principio fundamental detrás del Algoritmo de Quine-McCluskey es buscar la mayor cantidad de términos complementarios en la función para aumentar su complejidad.
c) El principio fundamental detrás del Algoritmo de Quine-McCluskey es dividir la función en segmentos iguales para simplificarla de manera eficiente.

39. **¿Cómo se compara el Algoritmo de Quine-McCluskey con el mapa de Karnaugh en términos de aplicabilidad y eficiencia?**
Respuesta:

a) El Algoritmo de Quine-McCluskey es más adecuado para funciones con un gran número de variables, mientras que el mapa de Karnaugh es más eficiente para funciones con pocas variables. (Correcta)
b) El Algoritmo de Quine-McCluskey es más adecuado para funciones con pocas variables, mientras que el mapa de Karnaugh es más eficiente para funciones con un gran número de variables.
c) El Algoritmo de Quine-McCluskey y el mapa de Karnaugh tienen la misma aplicabilidad y eficiencia en todos los casos.

40. ¿Cuál es la importancia de la simplificación de funciones booleanas en el diseño de circuitos lógicos?
Respuesta:

a) La simplificación de funciones booleanas es crucial para reducir la complejidad y el costo de los circuitos lógicos, lo que mejora su rendimiento y eficiencia. (Correcta)
b) La simplificación de funciones booleanas no tiene importancia en el diseño de circuitos lógicos, ya que no afecta la complejidad ni el costo.
c) La simplificación de funciones booleanas aumenta la complejidad y el costo de los circuitos lógicos, lo que mejora su rendimiento. y facilita

41. ¿Cuál es la función principal de una puerta lógica AND?
Respuesta
a) La función principal de una puerta lógica AND es realizar la operación de multiplicación lógica.
b) La función principal de una puerta lógica AND es realizar la operación de suma lógica.
c) La función principal de una puerta lógica AND es realizar la operación de negación lógica.

42. ¿Cómo se representa una puerta lógica OR en un circuito eléctrico?
Respuesta:
a) Una puerta lógica OR se representa mediante un símbolo con dos entradas y una salida que realiza la operación de suma lógica.
b) Una puerta lógica OR se representa mediante un símbolo con una entrada y una salida que realiza la operación de multiplicación lógica.
c) Una puerta lógica OR se representa mediante un símbolo con tres entradas y dos salidas.

43. **¿Cuál es la diferencia entre un circuito combinacional y un circuito secuencial?**
    a) Un circuito combinacional realiza operaciones lógicas basadas únicamente en las entradas actuales, mientras que un circuito secuencial tiene memoria y su salida depende de las entradas y el estado anterior.
    b) La diferencia entre un circuito combinacional y un circuito secuencial radica en la complejidad de las operaciones lógicas.
    c) Un circuito combinacional tiene memoria, mientras que un circuito secuencial opera solo con entradas actuales.

44. **¿Qué es una tabla de verdad en el contexto de circuitos lógicos?**
    Respuesta:
    a) Una tabla de verdad es una representación tabular que muestra todas las posibles combinaciones de entradas y las salidas correspondientes de una función booleana.
    b) Una tabla de verdad es una herramienta utilizada para programar microcontroladores.
    c) Una tabla de verdad es una representación gráfica de un circuito lógico.

45. **¿Por qué es importante la simplificación de expresiones booleanas en la electrónica digital?**

    Respuesta:
    a) La simplificación de expresiones booleanas reduce la complejidad de los circuitos y ahorra recursos, como tiempo y espacio.
    b) La simplificación de expresiones booleanas aumenta la complejidad de los circuitos y mejora su rendimiento.
    c) La simplificación de expresiones booleanas no tiene impacto en la electrónica digital.

46. **¿Qué son los mapas de Karnaugh y cómo se utilizan en la simplificación de funciones booleanas?**
    Respuesta:
    a) Los mapas de Karnaugh son herramientas gráficas que ayudan a simplificar funciones booleanas al identificar patrones de 1s en la tabla de verdad.
    b) Los mapas de Karnaugh son un tipo de dispositivo de almacenamiento de datos.
    c) Los mapas de Karnaugh son utilizados únicamente para aumentar la complejidad de las funciones booleanas.

47. ¿Cuál es la importancia de la minimización de funciones booleanas en el diseño de circuitos lógicos?

Respuesta:

a) La minimización de funciones booleanas es importante para reducir la complejidad y el costo de los circuitos lógicos, lo que mejora su rendimiento y eficiencia.
b) La minimización de funciones booleanas aumenta la complejidad de los circuitos lógicos y los hace menos eficientes.
c) La minimización de funciones booleanas solo es relevante en la programación de software.

48. **¿Cómo se utilizan los operadores lógicos AND, OR y NOT en el Algoritmo de Quine-McCluskey para simplificar funciones booleanas?**
Respuesta:

a) Los operadores lógicos AND se utilizan para identificar términos comunes en las mintérminos, los operadores OR se utilizan para combinar términos complementarios, y los operadores NOT se utilizan para invertir variables.
b) Los operadores lógicos AND se utilizan para invertir variables en el Algoritmo de Quine-McCluskey.
c) Los operadores lógicos AND se utilizan para combinar términos complementarios en el Algoritmo de Quine-McCluskey.

49. **¿Cuál es el papel de los operadores lógicos en la representación de funciones booleanas en el contexto del Algoritmo de Quine-McCluskey?**
Respuesta:

a) Los operadores lógicos son fundamentales para definir la relación entre las variables en una función booleana y permiten expresar las operaciones lógicas necesarias para la simplificación.
b) Los operadores lógicos no tienen ningún papel en la representación de funciones booleanas en el Algoritmo de Quine-McCluskey.
c) Los operadores lógicos solo se utilizan en la implementación de circuitos, no en la representación de funciones booleanas.

50. ¿En qué medida los operadores lógicos afectan la eficiencia y la complejidad de la simplificación de funciones booleanas utilizando el Algoritmo de Quine-McCluskey?

Respuesta:

a) Los operadores lógicos adecuados permiten simplificar funciones booleanas de manera eficiente y reducir la complejidad, mientras que un uso inadecuado puede aumentar la complejidad y dificultar la simplificación.

b) Los operadores lógicos no tienen ningún impacto en la eficiencia y la complejidad de la simplificación de funciones booleanas.

c) El uso de operadores lógicos siempre reduce la complejidad de la simplificación de funciones booleanas en el Algoritmo de Quine-McCluskey.

Bibliografía

Martín, Sergio. *Algoritmos alternativos para la simplificación de funciones booleanas*. UFG Editores. El Salvador. (2005).

Assandri, Armando. *Microprogramación*. Facultad de Ingeniería, Universidad Nacional de San Juan. Argentina. (2008).

García, Javier; Angulo Martinez, Ignacio; et Angulo Usategui, José. *Sistemas digitales y tecnología de computadores*. Thomson Ediciones. España. (2007).

Oliver, Joan; et Ferrer, Carles. *Diseño de sistemas digitales: introducción práctica*. Servei de Publicacions de la Universitat Autònoma de Barcelona. Cataluña. (1998).

Morris Mano, M. *Diseño Digital*. PEARSON Educación. Tercera edición. México. (2003)

Angulo, José. *ELECTRÓNICA DIGITAL MODERNA. Teoria y Práctica.* Editorial Paraninfo sa. 12ª edición. Madrid. (1994).

Falconi, Francisco. *TEMAS SELECTOS DE MATEMATICAS DISCRETAS*. México. (2013).

Mandado, Enrique. *SISTEMAS ELECTRONICOS DIGITALES*. MARCOMBO S.A. 8va Edición. España. (1998).

Holguín, Germán; et Holguín, Mauricio. *Principios y métodos combinatoriales en sistemas automáticos digitales.* Editorial Universidad Tecnológica de Pereira. Colombia. (2021).