INSTITUTO TECNOLÓGICO DE IZTAPALAPA

LOGICA COMBINACIONAL (FUNCIONAL)

LUGAR DE REALIZACIÓN:

INSTITUTO TECNOLOGICO DE IZTAPALAPA

Ing. Sistemas Computacionales

PRESENTA:

Hernández Ortiz Mauricio Javier
181080158
José Ángel Moreno Sánchez
181080341
Gómez García Francisco Javier
181080192
Padilla Castañeda Cesar Fabián
181080173

PROFESOR

M.Cs. Abiel Tomas Parra Hernández

Junio 6 de 2021

INDICE

INTRODUCCION

La lógica combinatoria es una teoría lógica elegante y poderosa que está conectada a muchas áreas de la lógica y ha encontrado aplicaciones en otras disciplinas, especialmente, en informática y matemáticas. CL se inventó originalmente como una continuación de la reducción del conjunto de constantes lógicas a un conjunto único en la lógica clásica de primer orden. Hablando filosóficamente, una expresión que no tiene variables limitadas representa la forma lógica de la fórmula original.

La sustitución es una operación crucial no solo en lógicas de primer orden, sino también en lógicas de orden superior , así como en otros sistemas formales que contienen un operador de vinculación de variables, como el λ-cálculos y el ε-cálculo. De hecho, llevar a cabo la sustitución correctamente es especialmente urgente λ-cálculos y en los lenguajes de programación funcional estrechamente relacionados. CL puede emular λ-abstracción a pesar de que CL no tiene operadores de vinculación de variables. Esto convierte a CL en un lenguaje de destino adecuado para la compilación de lenguajes de programación funcionales.

La conexión a λ Los cálculos podrían sugerir — correctamente — que CL es suficientemente expresivo para formalizar funciones recursivas (es decir, funciones computables) y aritmética. En consecuencia, CL es susceptible a los teoremas de incompletitud de tipo Gödel.

CL es un sistema de reescritura de términos arquetípicos (TRS). Estos sistemas comprenden una amplia gama de cálculos formales, desde especificaciones sintácticas de lenguajes de programación y gramáticas libres de contexto hasta algoritmos de Markov; incluso algunos problemas de teoría de números pueden verse como casos especiales de preguntas sobre TRS. Varias nociones y técnicas de prueba que se inventaron originalmente para CL, más tarde resultaron ser útiles en aplicaciones para TRS menos entendidos.

CL está conectado a lógicas no clásicas mediante escritura. Primero, se descubrió una correspondencia entre las fórmulas demostrables en el fragmento implicacional de la lógica intuicionista y los términos combinatorios tipificables. Luego, el isomorfismo se generalizó a otras bases combinatorias y lógicas implicacionales (como la lógica de la implicación relevante, la lógica lineal libre exponencial, la lógica afín, etc.).

Los factores de auto referencia en algunas paradojas, como la conocida paradoja del mentiroso y la paradoja de Russell. La comprensión teórica establecida de las funciones también desalienta la idea de auto aplicación. Por tanto, es notable que la CL pura no tipificada no excluya la auto aplicación de funciones. Además, sus modelos matemáticos mostraron que una teoría en la que las funciones pueden convertirse en sus propios argumentos es completamente sensible, además de consistente (lo que se estableció anteriormente usando métodos de teoría de la demostración).

RESUMEN

La lógica combinatoria es una rama de la lógica matemática que analiza ciertos procesos, como la sustitución, que están asociados con variables. Estos procesos se dan por sentados en la mayoría de las formulaciones de la lógica, pero son complejos, y dado que una parte fundamental de la teoría resultante es recursivamente indecidible, el análisis no es trivial. La lógica combinatoria contribuye a simplificar los fundamentos últimos de la lógica matemática y a explicar las paradojas; contiene una aritmética en la que exactamente aquellas funciones numéricas que son recursivas parciales son representables; y tiene aplicaciones potenciales para el estudio más profundo de áreas tales como cálculos lógicos de orden superior, programación de computadoras y lingüística.

Se denomina sistema combinacional o lógica combinacional a todo sistemadigital en el que sus salidas son función exclusiva del valor de sus entradas en unmomento dado, sin que intervengan en ningún caso estados anteriores de las entradas o de las salidas. Las funciones (OR, AND, NAND, XOR) son booleanas donde cada función se puede representar en una tabla de la verdad. Por tanto, carecen de memoria y de realimentación.

En electrónica digital la lógica combinacional está formada por ecuacionessimples a partir de las operaciones básicas del álgebra de Boole. Entre los circuitos combinacionales clásicos tenemos:

Lógicos: Generador/Detector de paridad

Multiplexor y De multiplexor Codificador y Decodificador

Conversor de código

Comparador

Aritméticos:

Sumador

Aritméticos y lógicos

Unidad aritmético lógica

Estos circuitos están compuestos únicamente por puertas lógicasinterconectadas entre sí.

OBJETIVOS

Objetivos generales

*Entender, analizar, diseñar e implementar circuitos, con lógica combinacional. Es importante en los procesos físicos y lógicos de la información, pues con base en esta comprensión es posible plantear soluciones integrales de la ingeniería informática y sus posibles aplicaciones en el día a día.

Objetivos específicos

*Familiarizar al estudiante con los circuitos básicos combinacionales y secuenciales, proporcionando las herramientas para el análisis y diseño de circuitos digitales, combinando compuertas y trabajando con máquinas de estado finito, al igual que mostrando ejercicios de algebra booleana junto a prácticas para que se comprenda mejor el funcionamiento de estos.

*Establecer los procedimientos para implementar circuitos usando compuertas, apoyándonos en la creación de circuitos con el programa multisim. El cual nos ayuda a poder ver en tiempo real como funciona cada compuerta ya sea sola o conectada con otras, el programa ofrece una interfaz sencilla, por lo que el manejo de este no tendría por qué tener problema.

*Diseñar sistemas de lógica combinacional, usando diversos procesos de diseño y análisis, apoyándonos en la simplificación de circuitos lógicos usando el método de mapas de Karnaugh.

*Conocer, entender y manejar los flip-flop y su tabla de verdad, así como sus ecuaciones para diseño, entender el cómo los flip-flop logran almacenar información y sus posibles estados de salida.

JUSTIFICACIÓN

Teniendo en cuenta que la única limitación de la aplicación del diseño y síntesis de circuitos digitales radica en la imaginación del diseñador, y que la mayoría de los procesos eléctricos y electrónicos se llevan a cabo en forma digital tanto en control como comunicaciones, electromedicina, procesamiento de imágenes etc., es obligatorio que el profesional en el área de la energía eléctrica moderno conozca las técnicas de modelado y síntesis de circuitos digitales de muy alto nivel de integración y velocidad, a través del lenguaje de descripción de hardware VHDL, para poder competir con éxito en el ambiente laboral.

Esta experiencia educativa es la primera del área de sistemas digitales. Fundamenta y define los conceptos que posteriormente serán extendidos a máquinas digitales algorítmicas para concretarse en dispositivos microcontroladores y microprocesadores. Plantea la descripción del sistema binario como una extensión de la representación de la notación decimal. Ofrece mecanismos para la conversión entre diferentes bases numéricas. También muestra la representación de información a partir de códigos binarios. Posteriormente se formaliza la presentación de los conceptos binarios, a partir de la representación que aporta el Álgebra de Boole, con sus axiomas, teoremas y postulados. Se ofrecen métodos de representación y simplificación de funciones a partir de procedimientos algebraicos, gráficos y tabulares. Se describen herramientas computacionales que auxilian en el diseño de sistemas electrónicos y son aplicadas a la descripción e implementación de sistemas digitales, así como los conceptos básicos del hardware de los PLD y sus técnicas de programación.

Los circuitos combinacionales al día de hoy se ocupan en muchos dispositivos ya sea como en centros comerciales con puertas automáticas que funcionan a la par con un sensor de movimiento, esto basándonos en los principios de la algebra booleana en la combinación de circuitos y la interpretación de numero binarios.

Este trabajo ayudará también a la mejor comprensión de los circuitos combinacionales usando como ejemplo una práctica sencilla en donde se ven diferentes circuitos conectados entre ellos y comprobar su función

Marco Teórico

Los Flip-Flops son los dispositivos con memoria más comúnmente utilizados. Sus características principales son:

Asumen solamente uno de dos posibles estados de salida.

Tienen un par de salidas que son complemento una de la otra.

Tienen una o más entradas que pueden causar que el estado del Flip-Flop cambie.

Sumadores:

Los sumadores son elementos de gran utilidad tanto en los sistemas de computación, como en el resto de la cotidianidad y la comprensión de su funcionamiento es fundamental en el estudio de los sistemas digitales.

Comparadores:

Los comparadores son circuitos combinacionales capaces de comparar dos combinaciones presentes en sus entradas indicando si son iguales o diferentes; en caso de ser diferentes, indican cuál de las dos es mayor. Tienen tres salidas que indican el resultado de la comparación: A=B, A<B y A>B.

Decodificadores:

Un decodificador es un circuito combinacional, su función es convertir un código binario de N bits de entrada y M líneas de salida (N puede ser cualquier entero y M es un entero menor o igual a 2N), de manera que cada línea de salida será activada para una sola de las combinaciones posibles de entrada.

Codificadores:

Un codificador es un circuito combinacional con 2^N entradas y N salidas, cuya misión es presentar en la salida el código binario correspondiente a la entrada activada. Existen dos tipos fundamentales de codificadores: codificadores sin prioridad y codificadores con prioridad.

Conversores de código:

Son circuitos combinacionales cuya función es cambiar los datos de un código binario a otro, esto es así porque para determinadas operaciones de transmisión y procesamiento de información son más eficaces unos códigos que otros. Se suelen implementar mediante dispositivos lógicos programables.

Multiplexores:

Un multiplexor básico posee varias líneas de entrada de datos y una única línea de salida, también posee entradas de selección de datos, que le permiten conmutar los datos digitales provenientes de cualquiera de las entradas y dirigirlos hacia la salida.

Demultiplexores:

Estos circuitos o dispositivos realizan la función contraria a los multiplexores, es decir, tiene una única entrada y 2n salidas con n señales de control. En realidad, uno cualquiera, ya sea el multiplexor o demultiplexor puede realizar las dos funciones.

Generador de paridad:

En los generadores de paridad la paridad puede ser par o impar. El bit de paridad se utiliza para detectar posibles errores en la transmisión del dato transmitido, mediante un comprobador de paridad que decepciona la información con el fin de validarla.

Circuito combinacional

Un circuito lógico digital es puramente combinacional si la salida del mismo, en un instante dado, depende única y exclusivamente del valor que tengan sus entradas en el momento considerado. En un circuito combinacional, salvo por el pequeño intervalo de tiempo que tarden en propagarse las señales, desde la entrada a la salida, dada la entrada, la salida estará determinada inmediata inmediatamente.

Mapas de Karnaugh

Un mapa de Karnaugh (también conocido como tabla de Karnaugh o diagrama de Veitch) es un diagrama utilizado para la simplificación de funciones algebraicas en forma canónica. A partir de la tabla de Karnaugh se puede obtener una forma canónica mínima (con el mínimo número de términos). En este texto emplearemos indistintamente los términos “mapa” y “tabla” de Karnaugh.

La tabla de Karnaugh consiste en una representación bidimensional de la función que se quiere simplificar. Si la función viene expresada como una tabla de verdad, entonces la tabla de Karnaugh puede verse como una forma alternativa de representación 2D. Puesto que la tabla de verdad de una función de n variables posee 2n filas, la tabla de Karnaugh correspondiente debe poseer también 2n celdas. La construcción de la tabla de Karnaugh pasa por codificar cada celda en código binario reflejado (o código Gray) de manera que celdas adyacentes tengan un código que difiere en un solo dígito.

Enfoques alternativos: Lógica básica y funtores predicados

En esta sección describimos brevemente dos ideas que están relacionadas con el trabajo de Schönfinkel o están motivadas por su uso de combinadores en la eliminación de variables ligadas.

La metalógica de Fitch

Desde finales de la década de 1930, Frederic Fitch trabajó en una lógica que llamó lógica básica. La etiqueta está motivada por su objetivo de proporcionar un marco en el que se pueda formalizar cualquier lógica. El enfoque de Fitch es completamente sintáctico (muy parecido al de Schönfinkel), y la "formalización" debe entenderse como la codificación de un sistema descrito formalmente en otro, no muy diferente de la aromatización de la sintaxis en el teorema de incompletitud de Gödel.

En 1942, Fitch introdujo una lógica que denominó A. Las expresiones enAse forman como términos combinatorios mediante una operación de aplicación binaria, que no se supone que sea asociativa. (Consulte la definición de términos combinatorios en la siguiente sección.) Sin embargo, las constantes deAno coinciden con las constantes de CL puro. Fitch usa 10 constantes: mí, o, EN, =, ∧, ∨, ES y ∗. Las primeras cinco constantes son combinadores, aunque la notación puede sugerir un significado diferente (informal). '='es la identidad sintáctica de las expresiones. '∧' y' están destinados a representar "y" y "o". ' E'es el análogo de Schönfinkel U, pero corresponde a un cuantificador existencial no vacío. Finalmente, '∗'es similar al operador de cierre transitivo para relaciones binarias o la estrella de Kleene. En particular, no hay negación o cuantificador universal en el sistema. Los usos de las constantes se caracterizan de la siguiente manera: algo así como los axiomas caracterizan a los combinadores.

1. $= ab$ si y solo si a y b son (sintácticamente) la misma expresión
2. εab si y solo si ba
3. $oabc$ si y solo si $a(bc)$
4. $\acute{\varepsilon}abc$ si y solo si bac
5. $\acute{o}abcd$ si y solo si $a(bc)d$
6. Wab si y solo si abb
7. $\wedge ab$ si y solo si a y b
8. $\vee ab$ si y solo si a o b
9. Eb si y solo si $\exists a.\ ba$
10. $*abc$ si y solo si abc y $\exists d.\ abd\&adc$

En CL, los axiomas son seguidos con nociones como un paso y reducción débil, la última de las cuales puede verse como un paso de cálculo o inferencia. (Consulte la siguiente sección para conocer algunas de estas nociones.) De manera similar, un cálculo axiomático para FOL, por ejemplo, contendría reglas de inferencia además de los axiomas. Uno de los obstáculos para penetrar las diversas presentaciones de la lógica básica es la falta de una formulación similar.

La lógica básica no se ha convertido (todavía) en un marco general ampliamente utilizado para la descripción de sistemas formales; sin embargo, Updike (2010) señala un renovado interés en este enfoque, que intenta situar la lógica básica en el contexto más amplio del trabajo fundacional a mediados del siglo XX.

La estrategia de eliminación de Quine

Desde finales de la década de 1930, W. V. O. Quine trabajó en una forma alternativa de eliminar las variables ligadas de la lógica de primer orden. Es plausible suponer que el objetivo de Schönfinkel era encontrar un solo operador en la lógica clásica y luego eliminar las variables ligadas, como afirma en Schönfinkel (1924), en lugar de definir un sistema simbólico general para describir todas las matemáticas. No obstante, CL pronto se fusionó con la lógica clásica de una manera más libre, lo que resultó en un sistema inconsistente.

Quine vio la manera de salir de una situación en la que la inconsistencia puede surgir a través de la tipificación implícita de constantes que son hasta cierto punto similares a los combinadores. Llamó a estos constantes funtores predicados e introdujo varios grupos de ellos, el último en Quine (1981).

Las presentaciones más comunes de FOL estipulan que un norte -place predicado seguido de una secuencia de norte términos (posiblemente, puntuados por comas y entre paréntesis) es una fórmula. (Esto contrasta con la visión de fórmulas de Schönfinkel y de acuerdo con las interpretaciones formales e informales de los predicados como norte-relaciones comerciales. En otras palabras, FOL no permite la "elaboración" de predicados o de sus interpretaciones). Quine se suscribe al punto de vista de que las secuencias de términos siguen a los predicados.

Términos combinatorios y sus principales propiedades

Reducción, igualdad y sus formalizaciones

Las paradojas que descubrieron Georg Cantor y Bertrand Russell a finales del siglo XIX y principios del XX implican la pertenencia a un conjunto por sí mismos. La teoría ramificada de tipos debida a Alfred N. Whitehead y Bertrand Russell, y ZF (la formalización de la teoría de conjuntos que lleva el nombre de Ernst Zermelo y Abraham A. Fraenkel) excluye la auto pertenencia. Sin embargo, parece que siempre ha existido el deseo de crear una teoría que permita la auto pertenencia o la auto aplicación. De hecho, una de las motivaciones de Curry para el desarrollo de CL fue el objetivo de construir un lenguaje formal que incluya una amplia gama de expresiones bien formadas, algunas de las cuales, bajo ciertas interpretaciones, pueden resultar sin sentido. (Esta idea puede compararse con la de von Neumann – Bernays – Gödel formalización de la teoría de conjuntos, en la que, sin el axioma de fundamento, se puede demostrar que la clase Russell no es un conjunto y, por tanto, una clase propiamente dicha).

Algunos ejemplos de lenguaje natural proporcionan una ilustración conveniente para aclarar la diferencia entre (1), que es una expresión bien formada (pero sin sentido) y (2), que es una oración significativa (pero mal formada). (El significado de (2), por supuesto, debe tomarse con un grano de sal. En realidad, Kurt Gödel demostró que el sistema de PM era incompleto en 1930. Por lo tanto, (2) puede adivinarse, utilizando pistas sintácticas y semánticas, Gödel demostró en 1930 que era una versión distorsionada de (2 ′) La aritmética de Peano estaba incompleta.)

(1) La derivada de $\lambda x\,(x^2 + 4x - 6)$ desea declarar que las funciones son inteligentes.
(2) La aritmética de Peano resulta incompleta con Gödel en 1930.

El uso de meta variables abarca la sustitución (que ilustramos anteriormente sobre el término EN METRO norte). El axioma de identidad y la regla de la transitividad implican que es una relación transitiva y reflexiva. Las dos últimas reglas caracterizan la aplicación como una operación que es monótona en ambos lugares de sus argumentos .CL incluye solo S y A, porque los otros combinadores se pueden definir a partir de ellos, como ya mencionamos en la sección 1.2, y como explicamos con más precisión hacia el final de esta sección.

La noción de reducción es una relación más débil que la reducción de un paso, por lo que es útil distinguir una subclase de términos utilizando la relación más fuerte. Un término está en forma normal (nf) cuando no contiene redexes. Tenga en cuenta que la reducción de un paso no necesita disminuir el número total de redexes que contiene un término, por lo tanto, no se sigue que cada término pueda convertirse en un término en nf a través de un número finito de reducciones de un paso. De hecho, algunos términos no se reducen a un término en nf.

La reducción es posiblemente una relación importante entre términos que denotan funciones. Los pasos típicos en la ejecución de un programa y en otros cálculos concretos son aplicaciones de funciones en lugar de movimientos en la otra dirección, lo que se llama expansión. Sin embargo, la noción de igualdad de funciones es familiar para todos desde las matemáticas, y la noción análoga también se ha introducido en CL. El cierre transitivo, reflexivo y simétrico de la relación de reducción de un paso se denomina igualdad (débil). Se puede obtener una formalización de CL ecuacional extendiendo la lógica ecuacional estándar con axiomas combinatorios y reglas que caracterizan las constantes combinatorias y la operación de aplicación.

Cálculo ecuacional para CL ($CL_{=}$).

$$METRO = METRO \qquad AMETRO norte = METRO \qquad SMETRO norte\, PAG = METRO\, PAG\, (norte\, PAG)$$

$$\frac{METRO = norte \quad norte = PAG}{METRO = PAG} \qquad \frac{METRO = norte}{norte = METRO} \qquad \frac{METRO = norte}{METRO\, PAG = norte\, PAG} \qquad \frac{METRO = norte}{PAG\, METRO = PAG\, norte}$$

Metodología de trabajo

Esta metodología consiste en la organización del trabajo diario en base a un panel de tareas. No propone cambios en las prácticas de ingeniería ni una nueva definición de proceso o estilo de trabajo. En cambio, se diseña para evitar la sobreproducción y para asegurarse de que los componentes pasan de un subproceso al siguiente en el orden adecuado.

Así se desarrolla un sistema de relleno que controla las cantidades producidas para reponer los componentes solo cuando sea necesario. Eso sí, en lugar de utilizar Kanban específicos, también se pueden poner en marcha otros sistemas reutilizables, como contenedores, palets o bandas codificadas. No obstante, una alternativa es la producción anticipada basándose en predicciones.

Elegimos la metodología Ágil Kanban porque en los sistemas combinacionales están formados por un conjunto de compuertas interconectadas cuya salida, en un momento dado ya que su función de la entrada es en ese mismo instante. aplicar el método Kanban en una empresa puede ayudar a mejorar sus procesos y su gestión de proyectos. Además de que su objetivo es gestionar de manera general cómo se van completando tareas y ayuda a evitar los dos problemas más importantes como cuellos de botella y tiempos muertos.

Al no contar con un software para la metodología Kanban el trabajo se fue desarrollando en tiempo real, a través de una videollamada, donde mostrábamos pantalla todos para poder ir monitoreando lo que se iba haciendo y poder detectar errores en el momento y así poder corregirlos de ser necesario, gracias a esto pudimos apoyar a nuestros compañeros cada que alguien tenía un inconveniente, se resolvía entre todos y así agilizar la finalización del proyecto, en caso de que se hiciera algún cambio individual , se notificaba con tiempo para así poderle dar seguimiento y determinar si se dejaba en el trabajo o no.

Desarrollo e implementación

1.- Un contactor R para el accionamiento de un motor eléctrico, está gobernado por la acción combinada de tres finales de carrera A, B y C. Para que el motor pueda funcionar, dichos finales de carrera deben reunir las siguientes condiciones:

1º) **A** accionado, **B** y **C** en reposo.

2º) **B** y **C** accionados, **A** en reposo.

3º) C accionado, **A** y **B** en reposo.

4º) A y **C** accionados, **B** en reposo.

Diseñar el circuito mínimo de puertas lógicas que cumple con dichas condiciones.

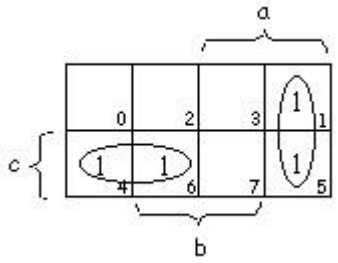
Tabla de verdad

c	b	a	R	
0	0	0	0	0
0	0	1	1	1
0	1	0	0	2
0	1	1	0	3
1	0	0	1	4
1	0	1	1	5
1	1	0	1	6
1	1	1	0	7

1ª FORMA CANONICA: Suma de productos cuya salida es igual a 1**:**

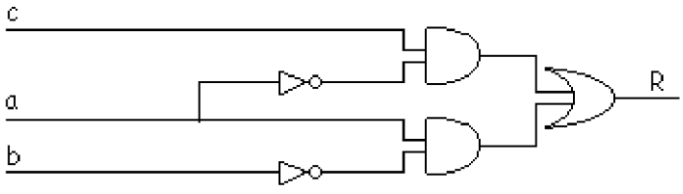
$$R = \Sigma_3 (1,4,5,6) \quad R = a \cdot \bar{b} \cdot \bar{c} + \bar{a} \cdot \bar{b} \cdot c + a \cdot \bar{b} \cdot c + \bar{a} \cdot b \cdot c$$

• Simplificación por Karnaugh:



$$R = a \cdot \bar{b} + \bar{a} \cdot c$$

- <u>CIRCUITO LÓGICO</u>:



2.- Imagina que tienes que diseñar una puerta electrónica para un garaje, de forma que solo debe abrirse cuando se pulse una determinada combinación de botones (A, B y C), según las condiciones indicadas. Diseña el circuito lógico que permita la apertura de la puerta del garaje, empleando las puertas lógicas que consideres oportuno.

Condiciones de apertura:

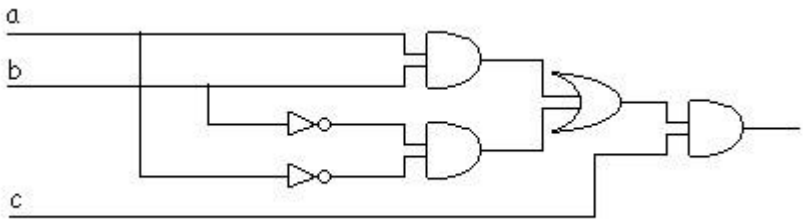
1) C pulsado, A y B en reposo. 2) A, B y C pulsados. Tabla de verdad

c	b	a	R	
0	0	0	0	0
0	0	1	0	1
0	1	0	0	2
0	1	1	0	3
1	0	0	1	4
1	0	1	0	5
1	1	0	0	6
1	1	1	1	7

1ª FORMA CANÓNICA: Suma de productos cuya salida es igual a 1

$S = \Sigma_3 (4,7) = \bar{a}\cdot b\cdot c + a\cdot b\cdot c = c\cdot(\bar{a}\cdot b + a\cdot \bar{b})$

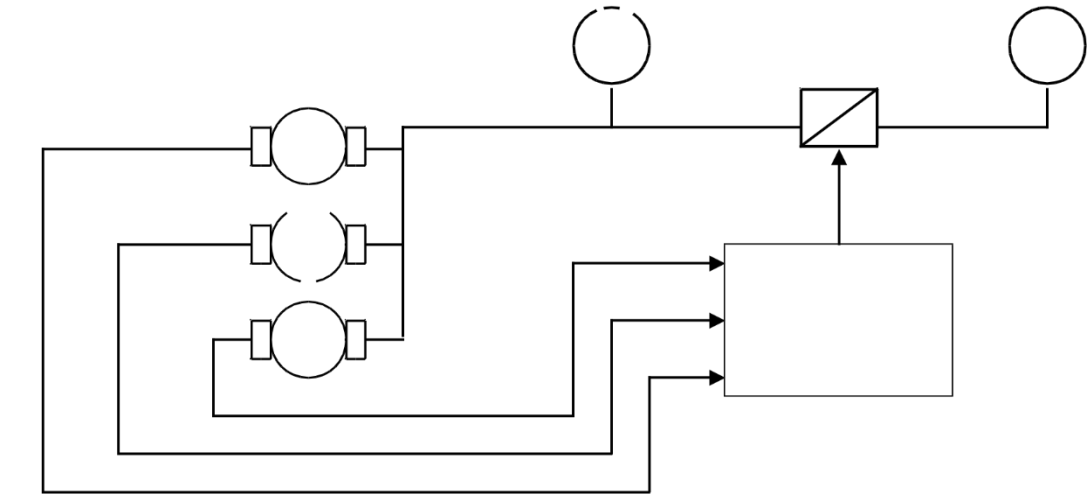
- CIRCUITO LÓGICO:



3.- En un determinado proceso industrial, disponemos de dos generadores de 15 Kw, cada uno, para alimentar a tres motores de 5 Kw, 10 Kw y 15 Kw, los cuales no funcionan siempre juntos (ver figura). Queremos realizar un automatismo que detecte los motores que están en funcionamiento en cada momento y ponga en marcha el segundo generador (G) cuando sea necesario.

SE PIDE:

1) La tabla de la verdad de la función G que controla el funcionamiento del 2º generador.
2) La función lógica en su primera forma canónica.
3) La expresión algebraica simplificada, obtenida mediante mapas de Karnaugh.
4) El circuito lógico correspondiente a la ecuación lógica sin simplificar y a la simplificada.



Motores - – **Generadores**

15 Kw - 15 Kw

G

A (5 Kw)

B (10 Kw)

C (15 Kw)

Automatismo

Entradas:	Salidas:
1 Motor funciona	**1 Funciona 2º generador**
0 Motor parado	**0 No funciona 2º gener**

- Tabla de la verdad:

Ocho combinaciones de entradas (2^3).
Salidas activas en función de las condiciones del enunciado.

A	B	C	G
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

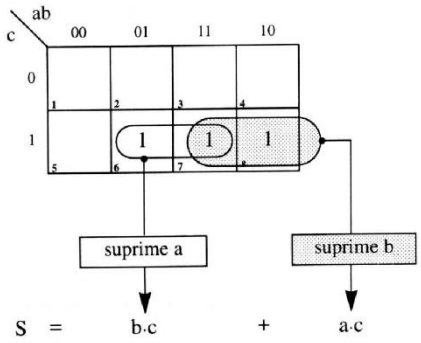
- Ecuación lógica:

  Primera forma canónica: Suma de productos lógicos que dan salida 1.

  $G = \Sigma_3(3,5,7)$

  $G = \overline{A} \cdot B \cdot C + A \cdot \overline{B} \cdot C + A \cdot B \cdot C$

- Simplificación por Karnaugh:



$G = B \cdot C + A \cdot C \rightarrow G = C \cdot (A+B)$

- Simplificación por Álgebra de Boole:

$G=A\cdot\overline{B}\cdot C+A\cdot B\cdot C+\overline{A}\cdot B\cdot C$ *Factor común A Cen2º y 3º miembros:

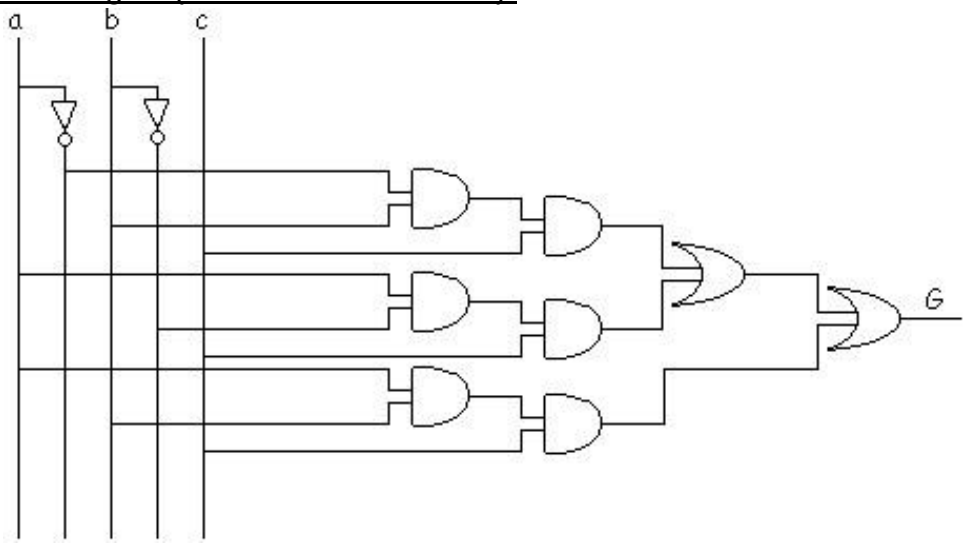
$G=A\cdot\overline{B}\cdot C+A\cdot C\cdot (B+\overline{B})$; * $(B+\overline{B}) = 1$ $G=A\cdot B\cdot C+A\cdot C$

*Factor común C en ambos términos:

$G=C\cdot(\overline{A}\cdot B+A)$; *Teorema: $\overline{A}\cdot B+A=A+B$

$G=C\cdot(A+B)$

- Circuito lógico (ecuación desarrollada):



- Circuito lógico (ecuación simplificada): Hay dos opciones

$G = B \cdot C + A \cdot C \rightarrow G = C \cdot (A + B)$

Resultados

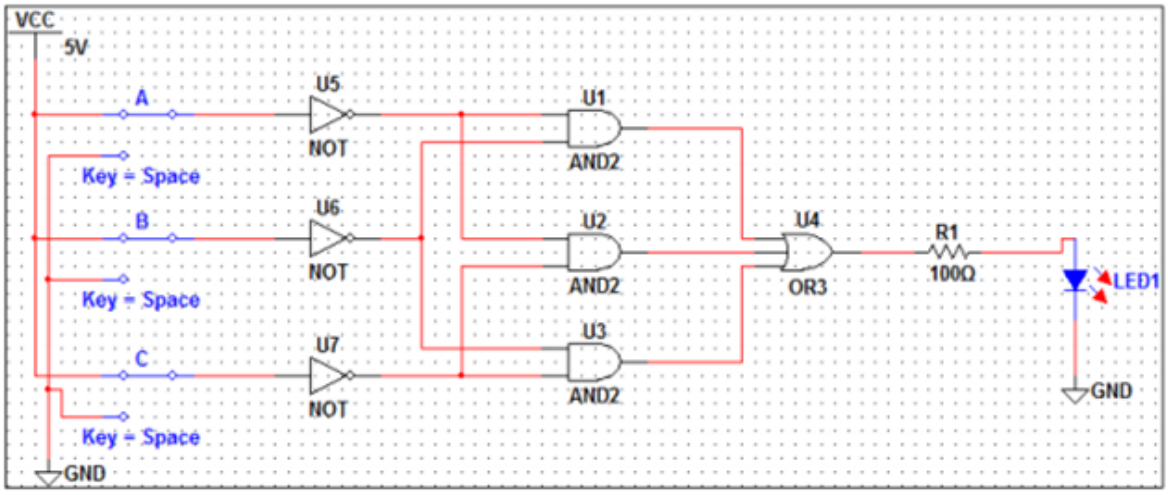
TEMA: Circuitos digitales combinacionales utilizando compuertas básicas.

OBJETIVOS: Diseñar un circuito digital que posea una salida y tres entradas, para que indique en la salida cuando dos o más entradas tengan un nivel lógico cero. Realizar el montaje y simulación del circuito lógico combinacional utilizando el software Multisim

MATERIALES:

Cantidad	Descripción	Característica, valor o serie
1	Computadora	Con el software Multisim
1	Diodos led	Rojo
1	Fuente de 5 V Cd / 1 A	5 V
3	Switch de 3 polos	SPDT
3	Compuertas NOT	7404
3	Compuertas AND	7408
1	Compuerta OR	7432
1	Resistor	330 Ω

ESQUEMAS:



SISTEMA CATEGORIAL Teoremas y postulados del álgebra de Boole: Primero se establece la relación de igualdad o equivalencia “=” para indicar que las dos variables x e y, son pertenecientes al conjunto B, son iguales; por ejemplo, x = y. Leyes de composición interna. En B se definen dos

leyes de composición interna, “+” (operador “O”, “OR”, o suma lógica) y “.” (Operador “Y”, “AND”, multiplicación o producto lógico); siendo B cerrado para estas operaciones.

$$\forall x \in B, \Longrightarrow a)\ x + y \in B$$
$$b)\ x\ .\ y \in B$$

Elementos neutros: Existen elementos neutros para ambas leyes de composición interna; las cuales son:

a) Elemento neutro para la suma $\exists\ 0 \epsilon\ B / \forall\ x \in B,\ x + 0 = 0 + x = x$

b) Elemento neutro para la multiplicación, $\exists 1\ \in \underset{\forall}{B} x \in B, x\ .\ 1 = 1\ .\ x = x$

Conmutatividad de las leyes de composición interna. La suma y la multiplicación lógica son conmutativas $; \forall x, y \in B$;

a) x + y = y + x

b) x y = y. x

Distributivita de las leyes de composición interna. En el álgebra de Boole la suma y la multiplicación son distributivas recíprocamente. $; \forall x, y \in B$;

a) $x + (y\ z) = (x + y)(x + z)$

b) $x\ (y + z) = x\ y + x\ z$

Todo elemento de B tiene su opuesto (o función NOT). A este elemento se le denomina inverso, opuesto, complemento o negado. Se representa de varias formas, dos de ellas son: (x, x’). La suma y el producto de una variable con su complemento dan como resultado “1” y “0” respectivamente $\forall x \in \mathbf{B}, \exists\ x \in \mathbf{B}/$

a) $x + x = 1$

b) $x.x = 0$

Teoremas:

Teorema de absorción (T1):

a) x + x. y = x

b) x (x + y) = x

Teorema (T2):

$\forall X, Y \in B$

$a) x \quad x + y \quad = x\, y$

$b) x + \ xy = x + y$

Teorema (T3):

$x = x$ Identidad

$x = x + 0$ Identidad de suma

$y = y$ Identidad

$y = y + 0$ Identidad de suma

Teorema (T4):

$a) x\, y + x\, y = x$

$b) \quad x + y \quad x + \ y \quad = x$

Teorema (T5):

$a) x\, y + x\, y\, z = x\, y + x\, z$

$b) \quad x + y \quad x + \ y + z \ = \ x + y \ (x + z)$

Teorema (T6):

$a) x + y = \ x\, .\, y$

$b) x\, .\, y \ = x + y$

DESARROLLO DE LA PRÁCTICA: Primero construya la tabla de verdad en correspondencia con el primer objetivo. Como se trata de tres variables, entonces el número de combinaciones será igual a 8 y la salida de la señal tendrá un uno lógico (F=1) cuando cumpla con la condición de dos o más entradas en bajo.

n	A	B	C	F
0	0	0	0	1
1	0	0	1	1
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	0
6	1	1	0	0
7	1	1	1	0

Una vez obtenida la tabla de verdad se seleccionan los minterms o maxterms para reducir a la mínima expresión y formar la función de conmutación. Debido a que la cantidad de maxterms y minterms son iguales se puede optar por cualquiera de los dos, por lo cual en este caso vamos a tomar los minterms:

$$F\ a,b,c\ = abc + \overline{a}bc + abc + abc$$ Forma canónica algebraica minterms.

A continuación se simplifica la función:

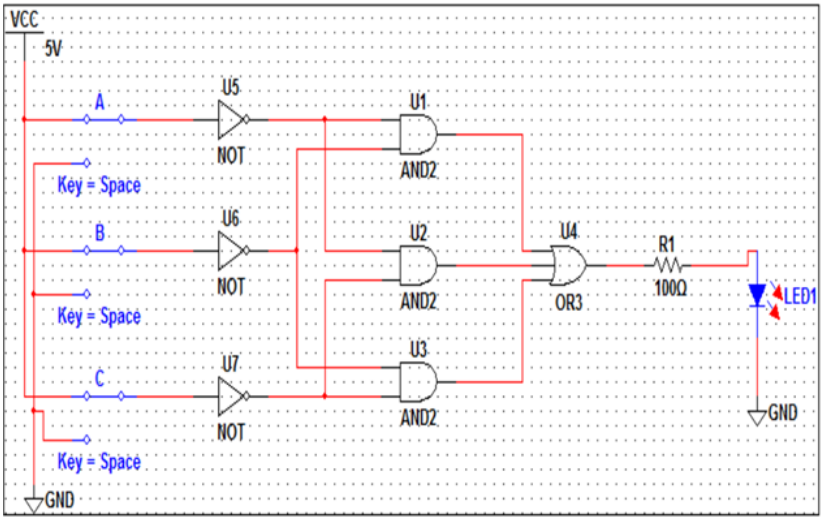
$$F\ a,b,c\ = abc + \overline{a}bc + abc + abc + abc + abc$$

$$F\ a,b,c\ = ab\ c + c\ + ac\ b + \mathrm{b}\ + bc\ (a + a)$$ Factor común

$$F\ a,b,c\ = ab\ 1\ + ac\ 1\ + bc\ (1)$$ Elemento opuesto

$$F\ a,b,c\ = ab\ + ac\ + bc$$ Función de conmutación simplificada

Proceda a realizar el esquema electrónico de la función obtenida, realice la simulación comprobando la tabla de verdad:



PREGUNTAS DE CONTROL: ¿De qué forma puede, el teorema de Morgan, ayudar a simplificar circuitos digitales? Es de mucha ayuda porque nos permite simplificar funciones muy grandes y convertirlas en funciones más pequeñas y comprensibles.

$$x + y = x.y \qquad (x + y) + x.y = 1$$

$$x.y = x + y \qquad x.y\ .\ x + y = 0$$

¿Cuál es el funcionamiento que cumple cada compuerta básica?

NOT Cumple la función de que todo nivel alto lo convierte en bajo y viceversa.

AND Cumple la función si o solo si las entradas son de nivel alto solo así la salida será alta caso contrario será bajo.

OR Cumple la función de dar un nivel alto sí una de las dos entradas es alta caso contrario será baja.

Conclusión

En conclusión, se pueden mencionar distintas cosas, en primer lugar, creemos que podemos experimentar y probar con distintas metodologías de trabajo, pero a nuestro parecer la ágil kanban fue la más efectiva para el equipo. en primer lugar, porque nosotros consideramos que somos concretos, intentamos llegar siempre al punto de la manera más fácil y rápida, es evidente que esto lleva mucho tiempo y borradores. Un punto en contra de esta metodología en función del equipo es que debido a que intentamos poner información justa nos auto generábamos ciertos retrasos, sin mencionar la cantidad de tecnicismos pendientes por definir para tener un mejor entendimiento como equipo, para después plasmarlo en este documento.

Logramos entender y analizar los circuitos con lógica combinacional, por otro lado, nos damos cuenta de que aún existe cierta dificultad al momento de diseñar e implementar los circuitos individualmente o como equipo, pero consideramos que es algo que con un poco más de practica y experiencia lograríamos.

Entender los ejercicios a pesar de que algunos los encontrábamos ya resueltos no fue tarea fácil, mucho menos entenderlos, una de las cosas que nos pasó es que aprendíamos un procedimiento, pero en los ejercicios venían con otro similar, aun así, teníamos muchas incógnitas con respecto a algunos puntos medios de la resolución.

Por otro lado, nos guiamos mucho con las palabras clave, si nos pasamos mucho tiempo definiendo ciertos términos, pero las palabras clave nos metían devuelta al camino.

En pocas palabras logramos entender que la lógica combinacional es una teoría de amplio espectro, ya que tiene diversos puntos de aplicación, sin embargo, tiene mayor aplicación en la informática y en las matemáticas, un circuito combinacional no cuenta con memoria ya que está conformada por un grupo de compuertas.

También podemos comprobar que los circuitos combinacional funcionan en base de la lógica combinacional implementando el uso de las tablas de verdad para rectificar como es su funcionamiento cuando están conectados entre sí.

No podemos atribuir todo el crédito al equipo, se necesita de explicación de un tutor o profesor que cuenta con más experiencia en el campo que nosotros, ya que al momento de querer entender algo no siempre se podrá al primer intento hasta que le preguntábamos al profe y el profe nos resolvía los inconvenientes y contratiempos que se nos iban presentando día con día a lo largo del semestre.

Fuentes de información

Abrigo, J. L. (2011). *SIMULACIÓN DE PRÁCTICAS DE ELECTRÓNICA*. Obtenido de PDF: https://dspace.unl.edu.ec/jspui/bitstream/123456789/18002/1/Balcazar%20Abrigo%2C%20Jorge%20Luis.pdf

Anonimo. (22 de Noviembre de 2017). *Lógica combinatoria*. Obtenido de Wikipedia: https://es.wikipedia.org/wiki/L%C3%B3gica_combinatoria

Anonimo. (2019). *COMBINATORY LOGIC*. Obtenido de enclyclopedia.com: https://www.encyclopedia.com/humanities/encyclopedias-almanacs-transcripts-and-maps/combinatory-logic

Carmen Baena, J. I. (1997). *SISTEMAS DIGITALES*. Obtenido de PDF: https://www.dte.us.es/docencia/eps/giei/ed/teoria/SistemasDigitales_Feb09.pdf

Castellanos, M. P. (s.f.). *TEMA 1. Sistemas Combinacionales*. Obtenido de PDF: https://www.cartagena99.com/recursos/alumnos/apuntes/tema_1_combinacionales.pdf

Farrugia, A. (s.f.). *Combinatory Logic: From Philosophy and Mathematics to Computer Science*. Obtenido de PDF: https://www.um.edu.mt/library/oar/bitstream/123456789/38118/1/Alexander%20Farrugia.pdf

Seldin, J. P. (16 de Noviembre de 2020). *Combinatory Logic*. Obtenido de Stanford Encyclopedia of Philosophy: https://plato.stanford.edu/entries/logic-combinatory/

Shabunin, L. (20 de Septiembre de 2014). *Combinatory logic*. Obtenido de Encyclopedia of Mathematics: https://encyclopediaofmath.org/wiki/Combinatory_logic

Anexo

Procedimiento de análisis.

El análisis de un circuito combinacional consiste en determinar la función que ejecuta el circuito. Se inicia con un diagrama del circuito lógico dado y culmina con un conjunto de ecuaciones booleanas o una tabla de verdad junto con una posible explicación de la operación del circuito.

Si el diagrama de lógica que se analiza se acompaña de un nombre de función o bien de un planteamiento de lo que se supone se realiza, entonces el problema de análisis se reduce a una verificación de la función expresada.

Para obtener las funciones booleanas de salida a partir de un diagrama de lógica se procede

de la manera siguiente:

1. Rotúlense todas las salidas de las compuertas que sean función únicamente de variables de entrada con símbolos arbitrarios. Determínense las funciones booleanas para estas compuertas.

2. Rotúlense las compuertas que sean función de variables de entrada y de las compuertas antes rotuladas, con diferentes símbolos arbitrarios. Determínense las funciones booleanas para estas compuertas.

3. Repítase el proceso que se describió en el paso 2 hasta que se obtengan las salidas del circuito en términos de las variables de entrada.

Procedimiento de diseño.

El diseño de circuitos combinatorios comienza desde la especificación del problema y culmina en un diagrama de circuitos lógicos o en un conjunto de funciones booleanas del cual se puede obtener el diagrama de lógica. En el procedimiento se aplican los pasos siguientes:

1. De las especificaciones del circuito, determinar el número requerido de entradas, salidas y asígnese un símbolo alfabético (o letra) a cada una.

2. Obtener la tabla de verdad que define la relación requerida entre entradas y salidas.

3. Defina las funciones booleanas simplificadas para cada salida como función de las variables de entrada.

4. Trácese el diagrama de lógica.

Decodificadores

Un decodificador n a 2n es una red lógica combinatoria de varias salidas, con líneas de entrada y 2n señales de salida. Para cada posible condición de entrada, n de entrada, una y sólo una se lo una señal de salida tendrá el valor lógico 1.