PRÁCTICA 2

ÁLGEBRA DE BOOLE

SOBRE LA PRÁCTICA

En esta práctica se plantean una serie de ejercicios y problemas que tienen como objetivo que el estudiante realice simplificaciones e implementaciones que implican el uso álgebra de Boole y de las principales puertas básicas que podemos emplear. Así mismo abordaremos la simplificación de funciones mediante tablas de Karnaugh y la conversión de expresiones a su forma dual.

Se deberá realizar una memoria de la práctica en el que aparezcan las operaciones y diseños realizados así como los resultados algebraicos, numéricos o gráficos obtenidos como solución a cada ejercicio. El método de presentación será a través del UACloud mediante la Entrega de Práctica que se habilitará para ello. El formato de presentación, para evitar problemas a la hora de la visualización, será preferentemente pdf. Se deben enviar junto con la memoria los diseños *.circ utilizados para resolver cada uno de los apartados. Para que sea posible la entrega, todo ello debe ir incluido en un único paquete comprimido.

REFERENCIAS

- T.L. Floyd. *Fundamentos de Sistemas Digitales,* 9ª Edición, Capítulo 3, “Puertas Lógicas”; secciones 3-1 a 3-6 y Capítulo 4, “Álgebra de Boole y Simplificación Lógica”, Secciones 4-1 a 4-11.
- C. Blanco. *Fundamentos de Electrónica Digital*. Capítulo 2. Algebra de Boole y Funciones Lógicas.
- Transparencias Tema 2 de Fundamentos de los Computadores “Álgebra de Boole”.

OBJETIVOS

Una vez realizada la práctica debemos ser capaces de:

- Simplificar funciones lógicas aplicando los postulados y teoremas del Álgebra de Boole.
- Obtener experimentalmente una tabla de verdad.
- Construir y utilizar tablas de Karnaugh para la simplificación de funciones.
- Combinar los elementos de una tabla de Karnaugh para obtener una expresión mínima.
- Transformar expresiones en forma de Suma de Productos a su forma dual en forma de Producto de Sumas.

INTRODUCCIÓN TEÓRICA

La lógica trabaja solamente con dos situaciones normales: el “1” lógico y el “0” lógico. Estas condiciones son las mismas que responder sí o no a una pregunta. También las podemos interpretar como un interruptor cerrado (1) o abierto (0); como que se ha producido un evento (1) o no (0), etc. En la lógica booleana el 1 y el 0 no representan números, sino situaciones. En la lógica positiva, el 1 se representa por el término nivel ALTO y el cero por nivel BAJO. En un circuito que opere con lógica positiva, el voltaje más positivo será el 1, mientras que el menos positivo (generalmente 0 V) será el 0.

Los símbolos de las principales puertas lógicas se muestran en la Figura 1. Las puertas AND, OR y NOT (o inversor) son las denominadas puertas básicas, si bien existen otros dos tipos muy importantes dentro del mundo del diseño: las puertas NAND y NOR. Su importancia radica en que podemos considerarlas como universales, ya que se pueden utilizar para sintetizar cualquier función lógica, incluyendo la AND, la OR y la NOT.

Frecuentemente también encontraremos otros dos tipos de puertas como son la OR exclusiva (abreviada como EXOR) y la NOR exclusiva (EXNOR). Utilizando los postulados del Álgebra de Boole cualquier expresión se puede reducir a su expresión más simple o cambiarla para conseguir una implementación más eficiente. Además, es muy fácil obtener la expresión estándar en suma de productos o productos de sumas y su implementación.

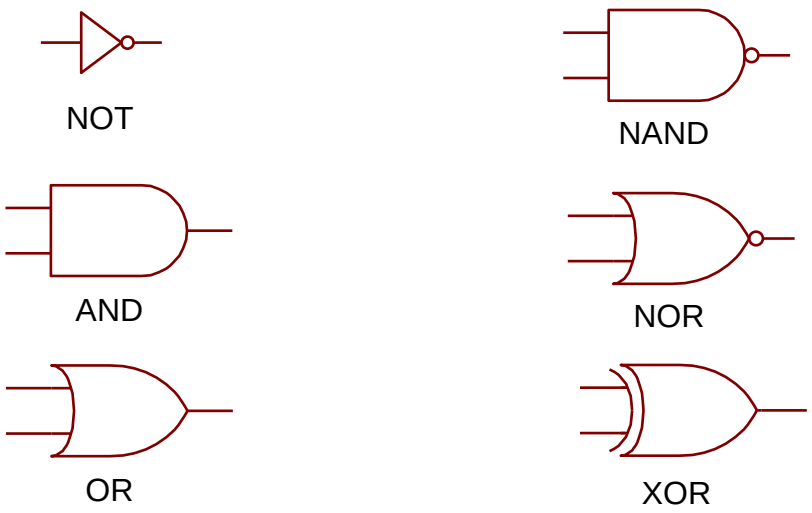

Figura 1. Símbolos de las principales puertas lógicas

Los circuitos deben materializarse a partir de la expresión simplificada de la función de salida obtenida a partir de la tabla de verdad.

Una de las técnicas más útiles de simplificación de circuitos lógicos combinacionales de que disponemos fue la desarrollada por M. Karnaugh en 1953. El método consiste en escribir la tabla de verdad dentro de una tabla en la que las celdas (cuadros) adyacentes difieren entre sí solamente en una variable (las celdas adyacentes poseen un borde común, bien sea horizontalmente o verticalmente). Cuando escribimos una tabla de Karnaugh, las variables se escriben siguiendo la secuencia del código Gray, tanto en la parte lateral como en la superior de la tabla. Los valores que adopta la salida aparecen con un ‘0’ ó ‘1’ en la celda correspondiente a su combinación de la tabla de verdad.

El tamaño de los grupos debe ser un entero potencia de 2 (es decir, 1, 2, 4, 8, etc.) y debe contener unos únicamente. Los grupos deben ser tan grandes como sea posible; todos los unos deben estar contenidos en algún grupo y pueden estar incluidos en más de uno si fuera necesario.

Una vez agrupados los unos en la tabla debemos determinar la función de salida. Cada grupo constituye un término producto de la función de salida simplificada. Las adyacencias dentro de cada grupo de celdas formado por más de un '1', nos permitirán eliminar de la expresión de salida las variables que cambien de valor entre las celdas agrupadas. Un grupo de dos unos adyacentes tendrá una sola adyacencia y sólo nos permitirá eliminar una variable. En un grupo de 4 unos podremos eliminar dos variables y en un grupo de 8, tres variables.

ELEMENTOS NECESARIOS

Para el desarrollo de la práctica utilizaremos el programa de simulación Logisim 2.7.1, que nos permitirá construir y analizar circuitos lógicos y también comprobar experimentalmente que los resultados obtenidos son los esperados.

REALIZACIÓN PRÁCTICA

1. Haciendo uso de puertas lógicas básicas (AND, OR, NOT), construye un circuito que implemente la función NOR Excusiva. $f(a,b)=\overline{a\oplus b}=ab+\bar{a}\bar{b}$. Utiliza los elementos "pin" para proporcionar las variables de entrada y el elemento "ver" visualizar la salida.

2. Dada la expresión $f=[a\bar{b}(c+\overline{b+d})+\overline{ab}]c\bar{d}$, simplifícala haciendo uso de los postulados, teoremas y leyes del álgebra de Boole que consideres oportunos. Implementa la expresión original y la simplificada y comprueba que sus tablas de verdad son coincidentes.

3. Una función lógica *f*(*a,b,c*) presenta la tabla de verdad de la Tabla 1.

Entradas				*Salida*
a	b	c	d	S
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

Tabla 1

a) Escribe su función algebraica en forma de suma de productos y de producto de sumas en su formato numérico.
b) Construye la tabla de Karnaugh asociada y simplifícala para obtener la expresión mínima en forma de suma de productos.
c) Vuelve a simplificar la tabla de Karnaugh para obtener la expresión en forma de producto de sumas.
d) Utilizando Logisim, implementa la función simplificada en los dos formatos y comprueba que las tablas de verdad coinciden con la original.

4. Sea la función $f = \prod_{4}(0,3,4,5,9,11,14)$.
    a. Simplifícala y obtén su expresión mínima en forma de suma de productos.
    b. Implementa el circuito haciendo uso del menor número de puertas básicas posible y obtén su tabla de verdad.
    c. Vuelve a implementar la función haciendo uso solamente de puertas NAND. Obtén su tabla de verdad y comprueba que la implementación es correcta.

5. Simplifica e implementa, haciendo uso del mínimo número de puertas, la función:

$$f = \sum_{5}(10,3,19,24,14,13,0,7,26,27) + \sum_{0}(15,25,31)$$

Comprueba que la salida del circuito es correcta para cada una de las combinaciones. ¿Qué ocurre con las indiferencias al obtener la tabla de verdad?