Clase 12

Algebra de Boole y operadores lógicos

12.1 Introducción

En la electrónica digital los circuitos lógicos tienen un papel imprescindible, dado que se ocupan de administrar la información de unos y ceros que se producen dentro de los circuitos, con niveles fijos de voltaje. Dentro de un circuito lógico podemos encontrar distintos elementos que lo forman, como las compuertas OR, AND, NOT, NOR, XOR, NAND, entre otras.

En concordancia con esto, el álgebra de Boole es una herramienta de fundamental importancia en el mundo de la computación. Las propiedades que se verifican en ella sirven de base al diseño y la construcción de las computadoras que trabajan con variables binarias (en las cuales los objetos básicos tienen solo 2 valores posibles en este caso 0 y 1) y por lo tanto esta álgebra es el fundamento teórico del funcionamiento de los circuitos lógicos.

12.2 El álgebra de Boole

Un sistema digital es aquel cuyos elementos son digitales, es decir, sólo pueden adoptar valores discretos. En la clase 3 vimos que la base 2, es decir el sistema binario, es el más adecuado desde el punto de vista de la confiabilidad y el costo.

Para poder analizar y construir un sistema de operaciones binarias para las computadoras se necesita tener en cuenta un álgebra binaria.

El Álgebra de George Boole, que data del año 1854, es sin dudas la más apropiada para este fin. La mayoría de los circuitos electrónicos, y de los sistemas de computación en general, tienen su origen en una función lógica. Pero esta puede ser bastante larga y compleja. Por eso George Boole (1815-1864) ideó un método para simplificar esa función lógica lo máximo posible, a través de ciertas reglas básicas o propiedades.

Claude Shannon en 1938 adaptó esta álgebra para la aplicación en sistemas digitales.

12.3 Postulados y teoremas del álgebra de Boole

Esta clase no se dedica exclusivamente al álgebra de Boole, para eso necesitaríamos varias clases, sin embargo, a fin de dar un sustento teórico al funcionamiento de las compuertas lógicas, veremos los principales postulados del álgebra de Boole, que son una serie de reglas que por definición deben cumplir todas las operaciones que realicemos.

En el álgebra de Boole utilizaremos los operadores booleanos, o funciones lógicas, OR y AND, equivalentes a la disyunción y a la conjunción. En esta álgebra las representaremos con los signos + y · respectivamente. Por lo tanto, el + es el OR, y el · es el AND.

En los siguientes postulados las letras a, b y c representan valores binarios, es decir cada una puede ser un cero o un uno.

Algo importante que a veces ayuda para comprender mejor esta álgebra es que el dígito binario 1 es equivalente al valor lógico de VERDADERO, y el dígito binario 0 es equivalente al valor lógico FALSO.

Valor binario	Valor lógico
1	VERDADERO
0	FALSO

1) Existencia de la identidad

Para cada operador existe el elemento identidad, que hace que un valor booleano a quede inalterable.

$a + 0 = a$

$a \cdot 1 = a$

2) Propiedad conmutativa para ambas operaciones

$a + b = b + a$

$a \cdot b = b \cdot a$

3) Propiedad distributiva para ambas operaciones

$a \cdot (b + c) = a \cdot b + a \cdot c$

$a + (b \cdot c) = (a + b) \cdot (a + c)$

4) Complemento o Inverso lógico

Todo elemento a tiene un inverso ~a (se lee no a; o a negado) tal que:

$a + \sim a = 1$

$a \cdot \sim a = 0$

12.4 Operadores booleanos básicos

Si A y B son expresiones o variables booleanas, se pueden generar nuevas expresiones a partir de ellas combinándolas con los llamados operadores booleanos.

En la siguiente tabla se muestran los símbolos originales de George Boole, pero como se ve en la columna de símbolos alternativos, hay distintas maneras de escribir los operadores booleanos básicos.

Nosotros usaremos los símbolos originales así como los alternativos AND, OR y NOT, ya que se usan para identificar las compuertas lógicas junto con otros operadores.

Los operadores booleanos básicos son los que aparecen en la siguiente tabla:

Operador booleano (nombre)	Símbolo	Símbolo alternativo
Producto lógico o conjunción	$\cdot$	$\wedge$ *AND*
Suma lógica o disyunción	$+$	$\vee$ *OR*
Complementación o negación	$\sim$	$\neg$ *NOT*

Vamos a explicar brevemente el significado de estos operadores.

Operador AND (·)

Este operador está representado por la multiplicación en el álgebra de Boole y equivale a la conjunción.

La traducción de AND en español es "y", lo que implica que la única manera de que el resultado de la operación sea 1, es que ambos operandos sean 1.

Visto de otra manera, para que el resultado sea verdadero ambos operandos deben ser verdaderos.

Ejemplos:

$0 \cdot 1 = 0$

$1 \cdot 1 = 1$

Operador OR (+)

Este operador está representado por la suma en el álgebra de Boole y equivale a la disyunción.

La traducción de OR en español es "o", lo que implica que la única manera de que el resultado de la operación sea 0, es que ambos operandos sean 0.

Visto de otra manera, para que el resultado sea falso ambos operandos deben ser falsos. Caso contrario es verdadero.

Ejemplos:

$$0 + 1 = 1$$

$$1 + 1 = 1$$

$$0 + 0 = 0$$

Operador NOT (~)

La compuerta NOT define la negación o inversión, es decir, su resultado es el opuesto del operando. A diferencia de las compuertas anteriores que pueden tener dos o más entradas, la compuerta NOT sólo posee una entrada. Si la entrada se encuentra en 1 entonces la salida será 0, y viceversa. Es decir:

$$\sim 1 = 0$$

$$\sim 0 = 1$$

12.5 Otros operadores booleanos

Introduciremos ahora, y explicaremos, los siguientes operadores booleanos: NOR, NAND, XOR, XNOR.

El operador NOR

El operador NOR es una combinación de un OR seguido por un NOT. Es decir, sería equivalente a realizar:

$$\sim(a + b)$$

Con lo cual sus resultados son los inversos de los del operador OR.

El operador NAND

El operador NAND es una combinación de un AND seguido por un NOT. Es decir, sería equivalente a realizar:

$$\sim(a \cdot b)$$

Con lo cual sus resultados son los inversos de los del operador AND.

El operador XOR

El operador XOR, también conocido como OR exclusivo, es una función lógica de dos variables de entrada, cuya salida será 1 sólo si una sola de sus entradas está en 1. Si las dos entradas son 1 su salida será cero, a diferencia del OR, el cuál sería 1.

Sería equivalente a:

$$(\sim a \cdot b) + (a \cdot \sim b)$$

El operador XNOR

Es un OR exclusivo seguido de un NOT. La función lógica que define es la siguiente:

$$\sim ((\sim a \cdot b) + (a \cdot \sim b))$$

Es decir, los resultados serán los opuestos a los del operador XOR.

12.6 Las tablas de verdad

La tabla de verdad de un operador es la muestra los resultados para las posibles entradas.

A continuación, mostraremos las tablas de valores para los tres operadores básicos.

En las columnas de las expresiones A y B aparecen los posibles pares de valores que pueden tomar.

En las columnas de las operaciones aparecen los resultados correspondientes.

Explicaré la tabla de verdad para el AND para que se comprenda mejor y luego se mostrarán las demás tablas, cualquier duda me consultan en el foro.

Tabla de verdad para AND

A	B	$A \cdot B$
1	1	1
1	0	0
0	1	0
0	0	0

Como observamos en el único caso en el que A·B da por resultado 1 es cuando ambos, A y B, son 1, tal como lo indica el operador AND. En cualquier otro caso el resultado es 0.

Visto de otro modo, como el operador AND significa "Y", entonces el resultado de A "Y" B es VERDADERO(1) si ambos A Y B son VERDADEROS(1).

Tabla de verdad para OR

A	B	$A+B$
1	1	1
1	0	1
0	1	1
0	0	0

Tabla de verdad para NOT

A	$\sim A$
1	0
0	1

Para el caso de las siguientes tablas la columna con la letra R representa el resultado del operador.

Tabla de verdad para NAND

A	B	R
0	0	1
0	1	1
1	0	1
1	1	0

Tabla de verdad para NOR

A	B	R
0	0	1
0	1	0
1	0	0
1	1	0

Tabla de verdad para XOR

A	B	R
0	0	0
0	1	1
1	0	1
1	1	0

Tabla de verdad para XOR

A	B	R
0	0	1
0	1	0
1	0	0
1	1	1

12.7 Funciones lógicas

A partir de los operadores lógicos básicos se pueden definir los del punto anterior, pero también se pueden definir otras funciones sin nombre específico.

Por ejemplo:

$$F = (a + b) \cdot \sim(a \cdot \sim b)$$

También se puede realizar la tabla de verdad para esta función F.

a	b	F
0	0	0
0	1	1
1	0	0
1	1	1

En estos casos lo más recomendable es ir analizando por partes. Por un lado, tenemos

$$(a + b)$$

Y por otro tenemos

$$\sim(a \cdot \sim b)$$

Y estas dos operaciones están unidas por un AND, es decir que para que F sea 1 ambas deben ser 1.

Para que (a+b) sea 1 deben ser alguna de las dos 1, por lo tanto el caso en el que a y b sean cero va a dar cero sí o sí.

Por lo tanto, nos quedamos con que o bien a o bien b tienen que ser 1, es decir, los casos 0 1, 1 0 y 1 1.

Nos queda analizar ahora en qué situación/es 1 la operación

$$\sim(a \cdot \sim b)$$

Como adelante tiene un NOT para que todo sea 1, tiene que ser 0 la operación

$$(a \cdot \sim b)$$

Y para que sea cero no tienen que ser 1 los dos, es decir a tiene que ser 0 y ~b tiene que ser 0.

Finalmente, para que ~b sea 0, b tiene que ser 1.

Por lo tanto, o bien a tiene que ser 0; o bien b tiene que ser 1.

Con lo cual para los casos 0 1 y 1 1 el resultado sería 1.

Entiendo que podría parecer complejo obtener la tabla de verdad. Voy a dejar en el foro de actividad de esta clase algunas tablas de verdad para que compartan la respuesta. La clave es ir analizando por partes.

12.8 Utilidad del álgebra de Boole en informática

Los operadores lógicos se implementan a nivel hardware mediante transistores y otros dispositivos electrónicos conformando lo que se denominan las compuertas lógicas y a través de varias compuertas lógicas se construyen los circuitos lógicos.

Ampliaremos este tema en la siguiente clase, pero sin ir más lejos, si recordamos la tabla de verdad del operador o compuerta XOR:

A	B	R
0	0	0
0	1	1
1	0	1
1	1	0

Observamos que los resultados concuerdan con la suma binaria, es decir:

0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

1 + 1 = 10

(no confundir este + con el operador OR, acá se está haciendo la suma binaria)

En el último caso se agregaría otra compuerta para implementar el 1 del acarreo.

A continuación, una imagen de la representación de la compuerta lógica XOR, junto a su tabla de verdad. A y B serían los bits de entrada y Q sería el bit de salida.