8. BOOLEAN ALGEBRAS

§8.1. Definition of a Boolean Algebra

There are many systems of interest to computing scientists that have a common underlying structure. It makes sense to describe such a mathematical structure abstractly and then theorems proved about it will be directly applicable in all these examples.

The structure is that of a **Boolean Algebra**. It is a set, with two binary operations + and ×, a unary operation $f(x) = \overline{x}$ and two distinct constants 0, 1 such that certain axioms hold. As in ordinary algebra, we write $a \times b$ as ab.

Axioms for a Boolean Algebra A:
(1) (Closure under Addition) For all $a \in A$, $a + b \in A$.
(2) (Associativity under Addition) For all $a, b, c \in A$, $a + (b + c) = (a + b) + c$.
(3) (Commutativity under Addition) For all $a, b \in A$, $a + b = b + a$.
(4) (Identity under Addition) For all $a \in A$, $a + 0 = a$.
(5) (Complement under Addition) For all $a \in A$, $a + \overline{a} = 1$.
(6) (Closure under Multiplication) For all $a \in A$, $ab \in A$.
(7) (Associativity under Multiplication) For all $a, b, c \in A$, $a(bc) = (ab)c$.
(8) (Commutativity under Multiplication) For all $a, b \in A$, $ab = ba$.
(9) (Identity under Multiplication) For all $a \in A$, $a1 = a$.
(10) (Complement under Multiplication) For all $a \in A$, $a\overline{a} = 0$.
(11) (Distributivity of Multiplication over Addition) For all $a, b, c \in A$, $a(b + c) = ab + ac$.
(12) (Distributivity of Addition over Multiplication) For all $a, b, c \in A$, $a + bc = (a + b)(a + c)$.

You will notice that most of these axioms hold in the ordinary algebra of real numbers. The exceptions are the Complement Laws.

§8.2. Examples of Boolean Algebras

Logic: This example is the smallest Boolean Algebra that can exist. Let $B_1 = \{0, 1\}$ and define addition, multiplication and complement as follows.

+	**0**	**1**
0	0	1
1	1	1

×	**0**	**1**
0	0	0
1	0	1

x	$\overline{\mathbf{x}}$
0	1
1	0

This elements in this algebra perform like the integers 0, 1 in ordinary arithmetic with the exception that 1 + 1 = 1, instead of 2. In fact these elements behave like the tags TRUE and FALSE, where 0 represents TRUE and 0 represents FALSE. Addition is the "or" operator, normally written as "$\vee$" and multiplication is the "and" operator, normally written as "$\wedge$", and $\overline{p}$ represents "not p". You should check that the 11 axioms hold for this system.

Sets: Let S be a set and let $B = \wp(S)$ be the power set. The elements of B are the subsets of S. The constants 0 and 1 represent the empty set and S itself. Addition represents the union of two sets, normally written as "$\cup$" and multiplication represents intersection, normally written as "$\cap$". If S is a finite set with n elements then B is a Boolean Algebra with 2^n elements.

Boolean Vectors: There are 2^n vectors $(x_1, x_2, \dots, x_n)$ where each $x_i \in B_1$. Let B_n be the set of all these vectors and define the operations of addition, multiplication and complement "pointwise". This means we define these operations on the functions in terms of the corresponding operations in B_1:

$$(x_1, x_2, \dots, x_n) + (y_1, y_2, \dots, y_n) = (x_1 + y_1,\ x_2 + y_2,\ \dots,\ x_n + y_n);$$
$$(x_1, x_2, \dots, x_n).(y_1, y_2, \dots, y_n) = (x_1 y_1,\ x_2 y_2,\ \dots,\ x_n y_n)\ ;$$
$$\overline{(x_1, x_2, \dots, x_n)} = (\overline{x_1}, \overline{x_2}, \dots, \overline{x_n})$$

§8.3. Properties of Boolean Algebras

Theorem 1: If x + y = 1 and xy = 0 then y = $\overline{x}$.
Proof: Suppose x + y = 1 and xy = 0. Then
y = y1 by Axiom 9
 = y(x + $\overline{x}$) by Axiom 5
 = y x + y $\overline{x}$ by Axiom 11
 = x y + y $\overline{x}$ by Axiom 8
 = 0 + y $\overline{x}$ by assumption
 = y $\overline{x}$ + 0 by Axiom 8
 = y $\overline{x}$ by Axiom 4
 = $\overline{x}$ y by Axiom 10
 = $\overline{x}$ y + 0 by Axiom 4
 = $\overline{x}$ y + x $\overline{x}$ by Axiom 10
 = $\overline{x}$ y + $\overline{x}$ x by Axiom 8
 = $\overline{x}$(y + x) by Axiom 11
 = $\overline{x}$(x + y) by Axiom 3
 = $\overline{x}$ 1 by assumption
 = $\overline{x}$ by Axiom 9

Corollary 1: For all x, $\overline{\overline{x}} = x$
Proof: The equations x + $\overline{x}$ = 1, x $\overline{x}$ = 0 not only say that $\overline{x}$ is the complement of x, but they also say that x is the complement of $\overline{x}$.

Corollary 2: $\overline{0} = 1$ and $\overline{1} = 0$.
Proof: 1 + 0 = 1 by Axiom 4 and
 1 0 = 0 by Axiom 9.
So 0 and 1 are complements of each other by Theorem 1.

Theorem 2: For all x, x 0 = 0.
Proof: x 0 = x 0 + 0 by Axiom 4
 = x 0 + x $\overline{x}$ by Axiom 10
 = x(0 + $\overline{x}$) by Axiom 11
 = x $\overline{x}$ by Axioms 3, 4
 = 0 by Axiom 10.

Theorem 3: For all x, x + 1 = 1.
Proof: $x + 1 = (x + 1)1$ by Axiom 9
= $(x + 1)(x + \overline{x})$ by Axiom 5
= $x + 1\overline{x}$ by Axiom 12

= $x + \bar{x}$ by Axiom 9
= 1 by Axiom 5.

Theorem 4 (De Morgan): For all x, y: $\overline{x + y} = \bar{x}\,\bar{y}$

Proof: We shall show that $\bar{x}\,\bar{y}$ (x + y) = 0 and $\bar{x}\,\bar{y}$ + (x + y) = 1. Then, by Theorem 1, it follows that $\bar{x}\,\bar{y}$ is the complement of x + y.

$\bar{x}\,\bar{y}$ (x + y) = ($\bar{x}\,\bar{y}$) x + ($\bar{x}\,\bar{y}$) y by Axiom 11
= $\bar{y}$ x $\bar{x}$ + $\bar{x}$ y $\bar{y}$ by Axioms 2 and 3
= $\bar{y}$ 0 + $\bar{x}$ 0 by Axiom 10
= 0 + 0 by Theorem 2
= 0 by Axiom 4.

$\bar{x}\,\bar{y}$ + (x + y) = (x + y) + $\bar{x}\,\bar{y}$ by Axioms 2, 3
= ((x + y) + $\bar{x}$)((x + y) + $\bar{y}$) by Axiom 12
= [y + (x + $\bar{x}$)] [x + (y + $\bar{y}$)] by Axioms 2, 3
= (y + 1)(x + 1) by Axiom 5
= 1 1 by Theorem 3
= 1 by Axiom 9.

Hence $\bar{x}\,\bar{y}$ is the complement of x + y.

Corollary: For all x, y $\overline{xy} = \bar{x} + \bar{y}$.

Proof: Substituting $\bar{x}$ for x and $\bar{y}$ for y in the theorem we get $\overline{\bar{x} + \bar{y}} = \bar{\bar{x}}\,\bar{\bar{y}} = xy$ by Theorem 1 (Cor 1). Hence xy and $\bar{x} + \bar{y}$ are complements of one another, that is, $\overline{xy} = \bar{x} + \bar{y}$.

§8.4. Symmetric Difference

We define a third binary operation in terms of the other two. We define

$$a \oplus b = a\,\bar{b} + \bar{a}\,b.$$

Theorem 5: For all a, b:

a ⊕ (b ⊕ c) = (a ⊕ b) ⊕c
a ⊕ 0 = a and
a ⊕ a = 0.

Proof: $a \oplus (b \oplus c) = a\,\overline{(b \oplus c)} + \bar{a}(b \oplus c)$

$= a\overline{(b\bar{c} + \bar{b}c)} + \bar{a}(b\bar{c} + \bar{b}c)$

$= a(\overline{b\bar{c}})(\overline{\bar{b}c}) + \bar{a}(b\bar{c} + \bar{b}c)$

$= a(\bar{b} + \bar{\bar{c}})(\bar{\bar{b}} + \bar{c}) + \bar{a}b\bar{c} + \bar{a}\bar{b}c$

$= a(\bar{b} + c)(b + \bar{c}) + \bar{a}b\bar{c} + \bar{a}\bar{b}c$

$= a\bar{b}b + a\bar{b}\bar{c} + acb + ac\bar{c} + \bar{a}b\bar{c} + \bar{a}\bar{b}c$

$= a\bar{b}\bar{c} + abc + \bar{a}b\bar{c} + \bar{a}\bar{b}c$.

$(a \oplus b) \oplus c = (a\bar{b} + \bar{a}b)\bar{c} + \overline{(a\bar{b} + \bar{a}b)}c$

$= a\bar{b}\bar{c} + \bar{a}b\bar{c} + \overline{a\bar{b}}\;\overline{\bar{a}b}\,c$

$= a\bar{b}\bar{c} + \bar{a}b\bar{c} + (\bar{a} + \bar{\bar{b}})(\bar{\bar{a}} + \bar{b})c$

$= a\bar{b}\bar{c} + \bar{a}b\bar{c} + (\bar{a} + b)(a + \bar{b})c$

$$= a\bar{b}\bar{c} + \bar{a}b\bar{c} + \bar{a}ac + \bar{a}\bar{b}c + bac + b\bar{b}c$$
$$= a\bar{b}\bar{c} + \bar{a}b\bar{c} + \bar{a}\bar{b}c + bac$$
$$= a\bar{b}\bar{c} + \bar{a}b\bar{c} + \bar{a}\bar{b}c + abc$$

$a \oplus 0 = \bar{a}0 + a\bar{0}$
$= 0 + a1$
= a.

$a \oplus a = a\bar{a} + \bar{a}a$
$= 0 + 0$
$= 0$.

These facts show that a Boolean Algebra is a group under this new operation. A **group** is an algebraic structure G with a binary operation $\circ$ that satisfies the properties:

(1) For all a, b $\in$ G, a $\oplus$ b $\in$ G.
(2) For all a, b, c $\in$ G, a $\oplus$ (b $\oplus$ c) = (a $\oplus$ b) $\oplus$c.
(3) There exists an element e $\in$ G such that a $\oplus$ e = a for all a $\in$ G.
(4) For all a $\in$ G there exists an element b such that a $\oplus$ b = e.

The group that arises in this way has two additional properties. Clearly a $\oplus$ b = b $\oplus$ a for all a, b $\in$ B. This is the commutative law, and a group that satisfies this property is called an Abelian group. Moreover a $\oplus$ a = 0 for all a $\in$ B. Using elementary group theory it follows that if B is finite, the number of elements of B is 2^n for some n, which is the size of B_n. In fact, all finite Boolean algebras have essentially the same structure as (in technical terms we say "are isomorphic to") B_n for some n.

§8.5. Duality

No doubt you noticed that the axioms for a Boolean Algebra came in pairs. If you swap + and $\times$, and swap 0 and 1, in one of each pair you will get the second. For example Axiom 11 states that for all a, b $\in$ B a(b + c) = ab + ac. Swapping + and $\times$ this gives a + bc = (a + b)(a + c) which is Axiom 12. Axiom 5 states that for all a $\in$ B, a + $\bar{a}$ = 1. Swapping +, and $\times$ and swapping 0 and 1, this becomes a $\bar{a}$ = 0 which is axiom 10.

The dual of a Boolean expression is obtained by swapping + and $\times$ and swapping 0 and 1. So axioms 6 to 10 are the duals of axioms 1 to 5 (and vice versa). Axioms 11 and 12 are duals of each other.

An important consequence of duality is the fact that any theorem in Boolean algebra remains a theorem if the expressions are replaced by their duals. Hence Boolean algebra theorems come in dual pairs.

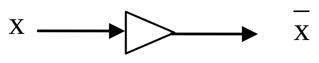
Principle of Duality: The dual of any theorem in Boolean algebra is also a theorem.
Proof: One can provide an independent proof of the dual by replacing every expression in the proof of the original theorem by its dual.

We have done this explicitly in several places. However it is simpler, having proved a theorem, to state its dual and to say "by the Principle of Duality".

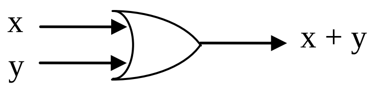
§8.6. Logic Gates

We can use Boolean algebra to describe an electronic circuit. Each component in such a circuit is called a gate. Each gate implements one of the three basic logic operations.

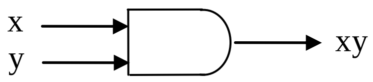
Inverter: This implements the complement operation.



OR gate: This implements the operation of addition.

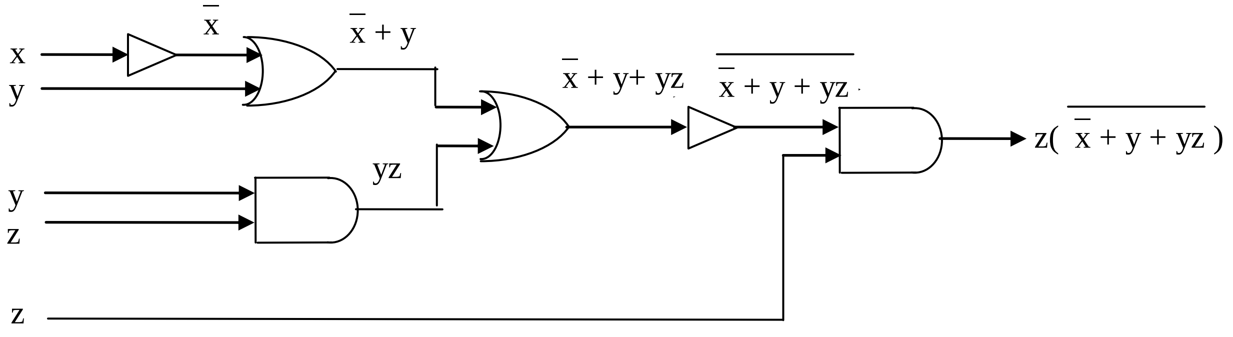


AND gate: This implements the operation of multiplication.



These gates can be combined to produce output according to any Boolean expression.

Example 1: The following circuit implements the function $z\left(\overline{\bar{x}+y+xz}\right)$.

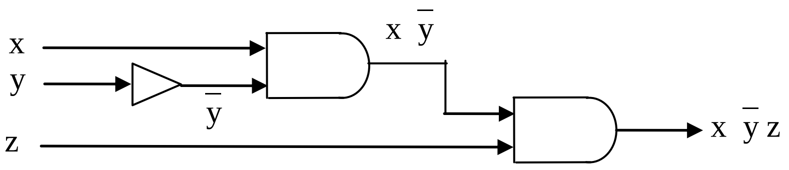


x
y
$\bar{x}$
$\bar{x}$ + y
$\bar{x}$ + y+ yz
$\overline{\bar{x} + y + yz}$
z($\overline{\bar{x} + y + yz}$)
y
z
yz
z

The function $y\left(\overline{\bar{x}+y+yz}\right)$ can be rewritten as follows.

$$
\begin{aligned}
z\left(\overline{\bar{x}+y+yz}\right) &= z\left(\bar{\bar{x}}\,\bar{y}\,\overline{yz}\right)\\
&= zx\bar{y}\left(\bar{y}+\bar{z}\right)\\
&= zx\bar{y}\,\bar{y}+zx\bar{y}\,\bar{z}\\
&= x\bar{y}z \text{ since } \bar{y}\,\bar{y} = \bar{y} \text{ and } z\bar{z} = 0.
\end{aligned}
$$

So a simpler circuit for this function is the following.

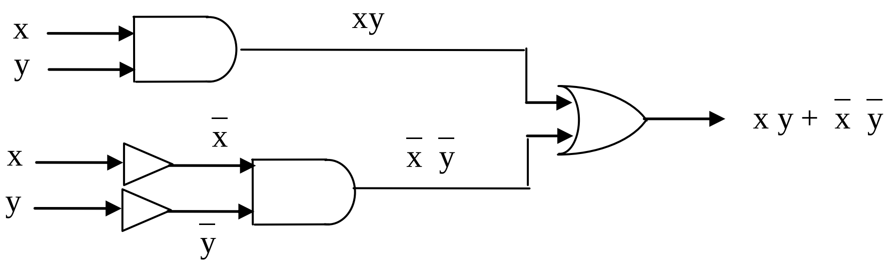


Example 2: Design a circuit that controls a light from two switches, so that changing the state of either of them changes the state of the light.
Solution: Let the two states of each switch be 0, representing OFF, and 1 representing ON. The output will be similarly 0 or 1. The function to be implemented by the circuit will be F(x, y). Suppose that F(1, 1) = 1, so that when both switches are ON the light is ON. If the first switch is switched OFF the light must go OFF so F(0, 1) = 0. If the second switch is now tripped the light must go ON again, so F(0, 0) = 1. Finally, F(1, 0) = 0.

So the Boolean function we must implement is:

x\y	**0**	**1**
0	1	0
1	0	1

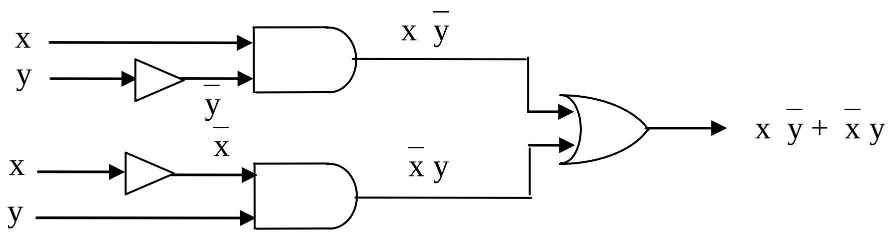
Clearly F(x, y) = x y + $\bar{x}\ \bar{y}$. A suitable circuit is the following.



Example 3: Design a circuit to add two integers modulo 2.
Solution: The function, F, to be implemented is the following.

x\y	**0**	**1**
0	0	1
1	1	0

Clearly F(x, y) = x $\bar{y}$ + $\bar{x}$ y. A suitable circuit is the following.



§8.7. Addition Circuits

Suppose we wish to design a circuit to add two bits, taking into account carrying as in a base 2 calculation. First, consider how integers, represented in binary, are added. For example, the calculation
6 + 7 = 13 in binary becomes:

```
  1 1 0
  1 1 1
1 1 0 1
```

We begin by adding the bits in the right-hand column: 0 + 1 = 1. There is no carry to the next column. Adding 1 + 1 in the next column we get 10, the binary equivalent of 2. We put down 0 in the second column and carry the 1. In the 3rd column from the right we must add 1 + 1 + 1, the third 1 being the 1 carried over from the previous column. This gives 3, which in binary is 11. So we put down one 1 and carry the other 1. The column to the left of this one has two blanks, effectively two zeros. So we add these two zeros, plus the carried 1, giving 1 in the left-most column.

So at each stage if the sum of the three inputs is 0 or 1 we write that down and the carry bit is 0. If the sum of the three inputs is 2 or 3 we write down 0 or 1 respectively and the carry bit is set to 1.

This circuit will have three inputs and two outputs. The third input and the second output is the "carry bit". The functions that lie behind this circuit are as follows.

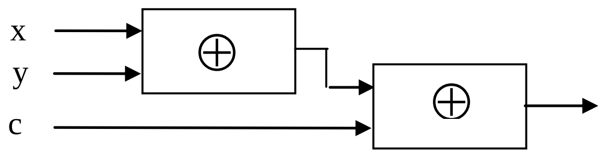
x	y	c	S(x, y, c)	C(x, y, c)
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

S(x, y, c) = $\bar{x}\,\bar{y}\,c + \bar{x}\,y\,\bar{c} + x\,\bar{y}\,\bar{c} + x\,y\,c$ and

C(x, y, c) = $\bar{x}\,y\,c + x\,\bar{y}\,c + x\,y\,\bar{c} + x\,y\,c$.

To implement this directly would require 3 inverters (one for each of x, y and c), 8 ADD gates (for the 8 products) and 6 OR gates for the sums. Let us see if we can simplify these functions first.

If we write addition modulo 2 as ⊕ then S(x, y, c) = $x \oplus y \oplus c = (x \oplus y) \oplus c$. So we can achieve this with two copies of the circuit in example 3.



Now C(x, y, c) = $\bar{x}\,y\,c + x\,\bar{y}\,c + x\,y\,\bar{c} + x\,y\,c$

= $\bar{x}\,y\,c + x\,\bar{y}\,c + x\,y(\bar{c} + c)$

= $\bar{x}\,y\,c + x\,\bar{y}\,c + x\,y$.

If two terms differ in just one factor we can always combine them in this way. But now comes the really tricky part.

C(x, y, c) = $\bar{x}\,y\,c + x\,\bar{y}\,c + x\,y$

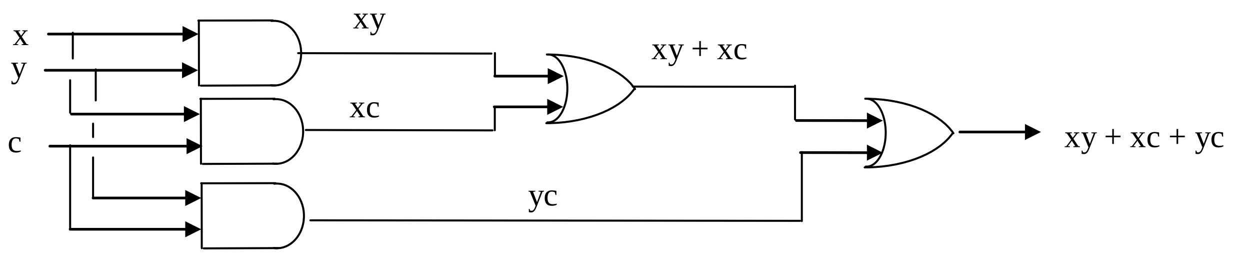
= $\bar{x}\,y\,c + x\,\bar{y}\,c + x\,y + x\,y\,c$.

Why? Because x y + x y c = x y(1 + c) = x y 1 = xy. But why would we want to add an extra term? Watch closely!

C(x, y, c) = $y\,c + x\,\bar{y}\,c + x\,y$, combining the first and last terms.

We are back to 3 terms, but we have two terms with 2 factors instead of only one, not to mention the fact that this will need one less inverter. Now we do it again.

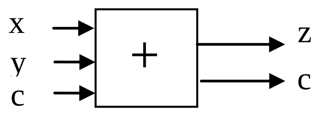
C(x, y, c) = $y\,c + x\,y\,c + x\,\bar{y}\,c + x\,y$ = yc + xc + xy, an expression that is considerably shorter than the one we started with. A suitable circuit to implement this is:



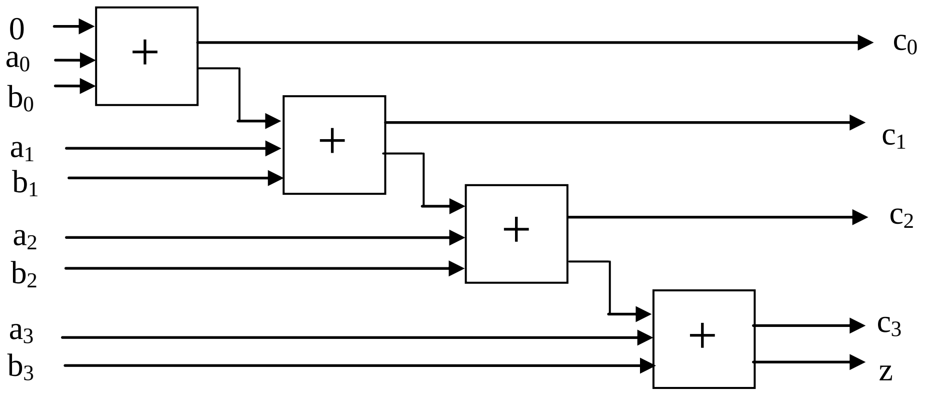
Each modulo 2 circuit has 2 inverters, 2 AND gates and 1 OR gate, so with two of them we have 4 inverters, 4 AND gates and 2 OR gates. Incorporating the carry circuit above we have 4 inverters, 7 AND gates and 4 OR gates, compared with 3 inverters, 8 AND gates and 6 OR gates if we had implemented it without simplification. This illustrates the usefulness of simplifying Boolean expressions before implementation.

Suppose now we wish to design a circuit in which numbers from 0 to 15 can be added. Such numbers can be represented by an 4 bit binary string of the form $b_3b_2b_1b_0$, ranging from 0000 to 1111.

Let us represent the above circuit as follows.



Here x, y are the bits being added, z is the bit that is output and c is the carry bit. We use 4 copies of this circuit.



§8.8. Disjunctive Normal Form

A Boolean expression can be extremely complicated, with many layers of complexity. Consider the following expression:

$$\overline{x+\overline{\left(y+\bar{z}\right)x}+\bar{z}y\overline{xz+y}}$$

We can simplify such an expression using three rules.

(1) $\overline{a+b}=\bar{a}\bar{b}$;

(2) $\overline{ab}=\bar{a}+\bar{b}$;

(3) $a(b+c)=ab+ac$.

The first two of these are the De Morgan Laws and the third is the Distributive Law.

Example 4: Simplify the expression $\overline{x+\overline{\left(y+\bar{z}\right)\bar{x}}+\bar{z}y\overline{xz+\bar{y}}}$.

Solution: $\overline{x+\overline{\left(y+\bar{z}\right)\bar{x}}+\bar{z}y\overline{xz+\bar{y}}}$ $=\bar{x}\left(y+\bar{z}\right)\bar{x}\left(\overline{\bar{z}y\overline{xz+\bar{y}}}\right)$

$=\bar{x}\left(y+\bar{z}\right)\bar{x}\left(z+\bar{y}+\left(xz+\bar{y}\right)\right)$

$=\bar{x}\left(y+\bar{z}\right)\left(z+\bar{y}+xz+\bar{y}\right)$

$=\left(\bar{x}y+\bar{x}\bar{z}\right)\left(z+\bar{y}+xz\right)$

$=\bar{x}yz+\bar{x}y\bar{y}+\bar{x}yxz+\bar{x}\bar{z}z+\bar{x}\bar{z}\bar{y}+\bar{x}\bar{z}xz$

$=\bar{x}yz+\bar{x}\bar{y}\bar{z}$.

A **literal** in a Boolean expression is one of the variables or its complement. So the literals for a binary expression $B(x_1, x_2, \ldots, x_n)$ are $x_1, x_2, \ldots, x_n$, $\bar{x}_1$, $\bar{x}_2$, … , $\bar{x}_n$, A **disjunctive normal form** in the variables $x_1, x_2, \dots, x_n$ is an expression as a sum of products where each term is a product of literals and where each variable occurs exactly once, either as itself or its complement. Each term gives exactly one combination of the variables under which the expression evaluates to 1.

Example 5: $x\,y + x\,x$ is not in disjunctive normal form since the term $x\,x$ contains x twice.
$x\,y\,z\,\bar{y}$ is not in disjunctive normal form since it contains both y and $\bar{y}$.
$x\,\bar{y}\,z + y$ is not in disjunctive normal form since x, z do not occur as factors of the term y.
$B(x, y, z) = x\,\bar{y}\,z + \bar{x}\,\bar{y}\,\bar{z} + x\,y\,\bar{z}$ is in disjunctive normal form. $B(x, y, z) = 1$ in exactly the following three instances:

$x = 1, y = 0, z = 1;$
$x = 0, y = 0, z = 1;$
$x = 1, y = 1, z = 0.$

Any Boolean expression can be expressed in disjunctive normal form. Since Boolean values commute under multiplication we insist that the variables occur in order in each term. If we also insist on the terms being arranged in "alphabetic order" we get a unique disjunctive normal form for each Boolean function. We do not always do this, but in counting disjunctive normal form we count expressions where the terms are merely rearranged as the same expression.

Theorem 6: Every Boolean function can be expressed uniquely in disjunctive normal form.

With n variables there are 2n literals and 2^n disjunctive normal forms. Each variable can either occur as itself or as its complement.

We can use the disjunctive normal form as the basis for the construction of a logic circuit, but it is generally not the simplest possible circuit. There are several criteria one might use to define "simplest". We may mean a circuit involving the fewest possible number of gates. This will potentially decrease the cost of manufacture. However there is a more important consideration.

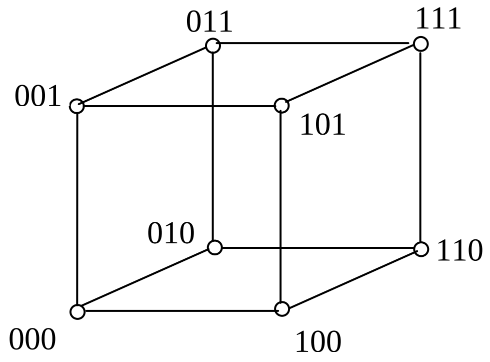
The speed of a circuit is dependent on its "depth". Many operations may occur in parallel, but the time taken for the whole operation depends on the largest number of steps that a given variable must pass.

§8.9. Representing of Boolean Expressions

We can express a disjunctive normal form in binary by coding variables as 1 and their complements as 0. If we arrange the variables in each term in a fixed order we can determine which variables are present as themselves, and which as their complements, by the positions of the 0's and 1's.

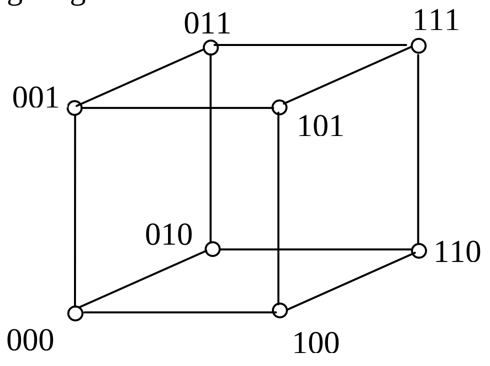
Example 6: The disjunctive normal form $x\,\bar{y}\,z + \bar{x}\,\bar{y}\,\bar{z} + x\,y\,\bar{z}$ can be coded as
101 + 000 + 110.

If there are three variables, as in this case, we can represent the possible terms by the vertices of a cube.



So a disjunctive normal form is represented by a set of these vertices.

Example 6 (continued): $B(x, y, z) = x\,\bar{y}\,z + \bar{x}\,\bar{y}\,\bar{z} + x\,y\,\bar{z}$ corresponds to the set of vertices coloured black in the following diagram.



Of course, if there are more than four variables we cannot use a geometric representation, at least not one we can draw, or even visualise. But mathematicians have no difficulty in discussing n-dimensional space, and even proving theorems about it, even though they are no more able to picture what is going on. They do this by working algebraically. After all, if points are described by their coordinates, there is no difficulty in considering the point (1, 0, 0, 1, 0) in 5-dimensional space. So we can say that a disjunctive normal form in n variables is represented by a set of points on a "hypercube", that is an n-dimensional version of a cube. Just do not expect to see it in a diagram!

Now a disjunctive normal form can often be simplified, with the simpler version still being a sum of products. It is just that some variables may be missing from sum terms.

Example 7: $x\,y\,z + x\,\bar{y}\,z + \bar{x}\,y\,z = x(y + \bar{y})z + \bar{x}\,y\,z = x\,z + \bar{x}\,y\,z$.

We code a sum of products by using the symbol "2" to represent a variable that is absent. It acts as a wildcard.

$\bar{x}$	0
x	1
variable x absent	2

Example 7 (continued): $x\,z + \bar{x}\,y\,z$ is represented by 121 + 011.
In fact, $x\,z + \bar{x}\,y\,z = (x + \bar{x}\,y)z$

$$= (x + \bar{x})(x + y) \text{ (by Axiom 12)}$$

$$= x + y.$$

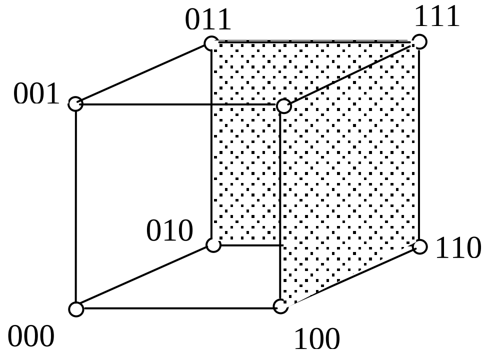
This is an even simpler form. It would be represented as 122 + 212.

Terms with wildcards are not represented geometrically as points, but rather as lines, or planes etc. Again, for illustrative purposes, we stick to three variables.

The term 121 is equivalent to 101 + 111. These are two adjacent points on the cube but we represent it as the line joining them rather than as the pair of points. Remember that only points on the cube have any significance so there is no point in considering any of the other points on this line.

The term 212 is represented by the relevant plane (with four points 010, 011, 110, 111).

Example 7 (continued):
The Boolean expression $x\,y\,z + x\,\bar{y}\,z + \bar{x}\,y\,z$ is simplified to $x + y$ which is coded as 122 + 212. Being of only three variables we can picture it as follows.



Do not confuse the Boolean expression "1" with the code "1". Since all variables are absent in "1" we write it as 222. However the Boolean expression "0" is difficult to represent in this way. Strictly speaking it is a sum of *no* terms. If we ever have occasion to represent it, and we are very unlikely to do so, we would give it the special symbol of ∅.

There are 27 strings of length 3 on the alphabet {0, 1, 2}. These correspond to the 8 vertices plus the 12 edges, plus the 6 faces plus the whole cube, making 27 geometric objects in all.

§8.10. Simplifying Boolean Expressions

Suppose we have a Boolean expression in disjunctive normal form. It can be simplified by use of the following operations.

(1) If two terms A, B are such that wherever they differ A has a "2" then B may be deleted.
This because of a repeated application of the identity $x + xy = x$.

Example 8: 21202 + 11200 = 21202.

(2) If two terms A, B differ in exactly one position, and neither symbol is "2", they may be combined by putting a "2" in that position.
This is because of the identity $x + \bar{x} = 1$.

Example 9: 1200 + 1210 = 1220.

(3) If two terms A, B differ in exactly two positions and the symbols in these positions in A are 2, a_2 and in B are b_1, b_2 with $b_2 < 2$ then B may changed by changing b_2 to 2.
This is because of the Boolean identity $x + \bar{x}\,y = x + y$. Note that the positions in which these differences occur need not be consecutive. Nor need they occur in the order of the variables. That is the a_1 and b_1 may occur for a "later" variable then the a_2 and b_2.

Example 10: 2 1 1 **2** 0 **1** + 2 1 1 **0** 0 **0** = 2 1 1 2 0 1 + 2 1 1 0 0 2.
Here A = 2 1 1 2 0 1, B = 2 1 1 0 0 0, $a_1 = 2$, $b_1 = 0$, $a_2 = 1$, $b_2 = 0$.

(4) If three terms A, B, C coincide except for three positions and the symbols in these positions are:

A: 2, $\bar{b}$, $\bar{c}$
B: a, 2, c
C: a, b, 2 where a, b and c are all < 2

then B and C may be combined by replacing b and c by 2.

Example 11: Simplify $B(u, v, w, x, y, z) = u\,v\,x\,\overline{y}\,z + u\,v\,w\,x\,\overline{z} + u\,v\,w\,x\,y + u\,\overline{v}\,w\,x$.
Solution: Coding this expression we get 112101 + 111120 + 111112 + 101122
= 112101 + 111122 + 101122 by rule 4
= 112101 + 121122 by rule 2.

Example 12: Simplify $B(x, y, z) = x\,y\,z + x\,\overline{y}\,z + \overline{x}\,y\,z + \overline{x}\,\overline{y}\,z + \overline{x}\,\overline{y}\,\overline{z}$.
Solution: We code it as 111 + 101 + 011 + 001 + 000
= 121 + 021 + 000 by rule 2 applied twice
= 221 + 000 by rule 2 again
Thus $B(x, y, z) = z + \overline{x}\,\overline{y}\,\overline{z}$.

Example 13: Simplify $B(a, b, c, d) = a\,b\,c\,\overline{d} + a\,\overline{b}\,c\,d + a\,\overline{b}\,c\,\overline{d} + \overline{a}\,b\,c\,d + \overline{a}\,\overline{b}\,c\,d + \overline{a}\,\overline{b}\,\overline{c}\,d$.
Solution: In code it becomes 1110 + 1011 + 1010 + 0111 + 0101 + 0011 + 0001
= 1210 + 1011 + 0111 + 0101 + 0011 + 0001 by rule 2
= 1210 + 2011 + 0111 + 0101 + 0001
= 1210 + 2011 + 0121 + 0001
= 1210 + 2011 + 0121 + 0201
= 1210 + 2011 + 0121 + 0201
= 1210 + 2011 + 0221
So $B(a, b, c, d) = a\,c\,\overline{d} + \overline{b}\,c\,d + \overline{a}\,d$.

These rules will simplify many Boolean expressions, but may not produce the shortest sum of products. There are algorithms which guarantee that the final answer will be the shortest possible for a sum of products.

The method of Karnaugh maps uses a version of the geometric representation while the Kline-McClusky method uses symbol manipulation along the lines of the above discussion. We do not discuss these here.