Binary Arithmetic

1. Binary Addition :

Case	A + B	Sum	Carry
1	0 + 0	0	0
2	0 + 1	1	0
3	1 + 0	1	0
4	1 + 1	0	1

0011010 + 001100 = 00100110

```
   1 1          carry
 0 0 1 1 0 1 0  = 26₁₀
+0 0 0 1 1 0 0  = 12₁₀
---------------
 0 1 0 0 1 1 0  = 38₁₀
```

2. Binary Subtraction :

Case	A - B	Subtract	Borrow
1	0 - 0	0	0
2	1 - 0	1	0
3	1 - 1	0	0
4	0 - 1	0	1

0011010 - 001100 = 00001110

1 1 borrow

0 0 ~~1 1~~ 0 1 0 = 26_{10}

- 0 0 0 1 1 0 0 = 12_{10}

0 0 0 1 1 1 0 = 14_{10}

3. Binary Multiplication:

Case	A x B	Multiplication
1	0 x 0	0
2	0 x 1	0
3	1 x 0	0
4	1 x 1	1

Example:

0011010 x 001100 = 100111000

$$
\begin{array}{rl}
0\,0\,1\,1\,0\,1\,0 & = 26_{10} \\
\times\, 0\,0\,0\,1\,1\,0\,0 & = 12_{10} \\
\hline
0\,0\,0\,0\,0\,0\,0 & \\
0\,0\,0\,0\,0\,0\,0 & \\
0\,0\,1\,1\,0\,1\,0 & \\
0\,0\,1\,1\,0\,1\,0 & \\
\hline
0\,1\,0\,0\,1\,1\,1\,0\,0\,0 & = 312_{10}
\end{array}
$$

Boolean Algebra

- Boolean algebra is the category of algebra in which the variable's values are the truth values, true and false, ordinarily denoted 1 and 0 respectively.
- It is used to analyze and simplify digital circuits or digital gates. It is also called Binary Algebra or logical Algebra.
- The important operations performed in Boolean algebra are – conjunction (∧), disjunction (∨) and negation (¬).

Suppose A and B are two Boolean variables, we can define the three operations as;

- A conjunction B or A AND B, satisfies A ∧ B = True, if A = B = True or else A ∧ B = False.
- A disjunction B or A OR B, satisfies A ∨ B = False, if A = B = False, else A ∨ B = True.
- Negation A or ¬A satisfies ¬A = False, if A = True and ¬A = True if A = False

Boolean Expression

- A logical statement that results in a Boolean value, either be True or False, is a Boolean expression. Sometimes, synonyms are used to express the statement such as ‘Yes’ for ‘True’ and ‘No’ for ‘False’. Also, 1 and 0 are used for digital circuits for True and False, respectively.
- Boolean expressions are the statements that use logical operators, i.e., AND, OR, XOR and NOT. Thus, if we write X AND Y = True, then it is a Boolean expression.

Boolean Algebra Terminologies

- Boolean Algebra: Boolean algebra is the branch of algebra that deals with logical operations and binary variables.

- Boolean Variables: A Boolean variable is defined as a variable or a symbol defined as a variable or a symbol, generally an alphabet that represents the logical quantities such as 0 or 1.

- Boolean Function: A Boolean function consists of binary variables, logical operators, constants such as 0 and 1, equal to the operator, and the parenthesis symbols.

- Literal: A literal may be a variable or a complement of a variable.

Complement: The complement is defined as the inverse of a variable, which is represented by a bar over the variable.

Truth Table: The truth table is a table that gives all the possible values of logical variables and the combination of the variables.

It is possible to convert the Boolean equation into a truth table.

The number of rows in the truth table should be equal to 2n, where “n” is the number of variables in the equation.

For example, if a Boolean equation consists of 3 variables, then the number of rows in the truth table is 8. (i.e.,) 23 = 8.

Boolean Algebra Rules

- Following are the important rules used in Boolean algebra.
- Variable used can have only two values. Binary 1 for HIGH and Binary 0 for LOW.
- The complement of a variable is represented by an overbar
- OR-ing of the variables is represented by a plus (+) sign between them. For example, the OR-ing of A, B, and C is represented as A + B + C.
- Logical AND-ing of the two or more variables is represented by writing a dot between them, such as A.B.C. Sometimes, the dot may be omitted like ABC.

Laws of Boolean Algebra

There are six types of Boolean algebra laws. They are:

- Commutative law
- Associative law
- Distributive law
- AND law
- OR law
- Inversion law

- **Commutative Law :**

Any binary operation which satisfies the following expression is referred to as a commutative operation. Commutative law states that changing the sequence of the variables does not have any effect on the output of a logic circuit.

A. B = B. A

A + B = B + A

- **Associative Law**

It states that the order in which the logic operations are performed is irrelevant as their effect is the same.

(A. B). C = A . (B . C)

(A + B) + C = A + (B + C)

- **Distributive Law**

Distributive law states the following conditions:

- A. (B + C) = (A. B) + (A. C)
- A + (B. C) = (A + B) . (A + C)

- **AND Law**

These laws use the AND operation. Therefore they are called AND laws.

A .0 = 0

A . 1 = A

A. A = A

- **OR Law**

These laws use the OR operation. Therefore they are called OR laws.

A + 0 = A

A + 1 = 1

A + A = A

- **Inversion Law**

In Boolean algebra, the inversion law states that double inversion of variable results in the original variable itself.

Boolean Algebra Theorems

De Morgan’s First law and De Morgan’s second law :

- These two theorems are used to change the Boolean expression.
- This theorem basically helps to reduce the given Boolean expression in the simplified form.
- These two De Morgan’s laws are used to change the expression from one form to another form.

1. **De Morgan’s First Law:**

De Morgan’s First Law states that (A.B)’ = A’+B’.
The first law states that the complement of the product of the variables is equal to the sum of their individual complements of a variable.

De Morgan’s Second Law:

- De Morgan’s Second law states that (A+B)’ = A’. B’.
- The second law states that the complement of the sum of variables is equal to the product of their individual complements of a variable.

TRUTH TABLE FOR DE MORGANS

A	B	A.B	$\overline{A.B}$	$\overline{A}$	$\overline{B}$	$\overline{A}+\overline{B}$
0	0	0	1	1	1	1
0	1	0	1	1	0	1
1	0	0	1	0	1	1
1	1	1	0	0	0	0

A	B	$\overline{A+B}$	$\overline{A}$	$\overline{B}$	$\overline{A}.\overline{B}$
0	0	1	1	1	1
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	0

- Representation of Boolean expression can be primarily done in two ways. They are as follows:

Sum of Products (SOP) form (A'.B'.C + A'.B.C + A.B.C')

Product of Sums (POS) form

Note:

If the number of input variables are n, then the total number of combinations in Boolean algebra is 2^n.

Sum of Products (SOP): It is one of the ways of writing a boolean expression.

it is formed by adding (OR operation) the product terms.

These product terms are also called as 'min-terms'.

Min-terms are represented with 'm', they are the product(AND operation) of boolean variables either in normal form or complemented form.

Product of Sums (POS): **POS is product of maxterms**

As the name suggests, it is formed by multiplying(AND operation) the sum terms.

These sum terms are also called as ‘max-terms’.

Max-terms are represented with ‘M’, they are the sum (OR operation) of Boolean variables either in normal form or complemented form.

(A+B+C).(A+B'+C).(A'+B+C).(A'+B+C').(A'+B'+C')

Consensus Theorem

Redundancy theorem is used as a Boolean algebra trick in Digital Electronics. It is also known as Consensus Theorem:

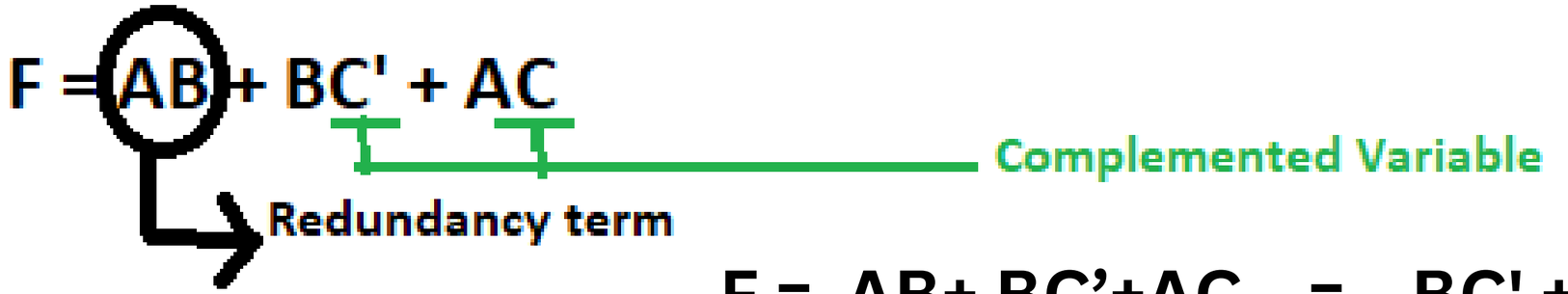
AB + A'C + BC = AB + A'C

(A+B).(A'+C).(B+C) = (A+B).(A'+C)

following conditions must be met before using the Redundancy theorem:

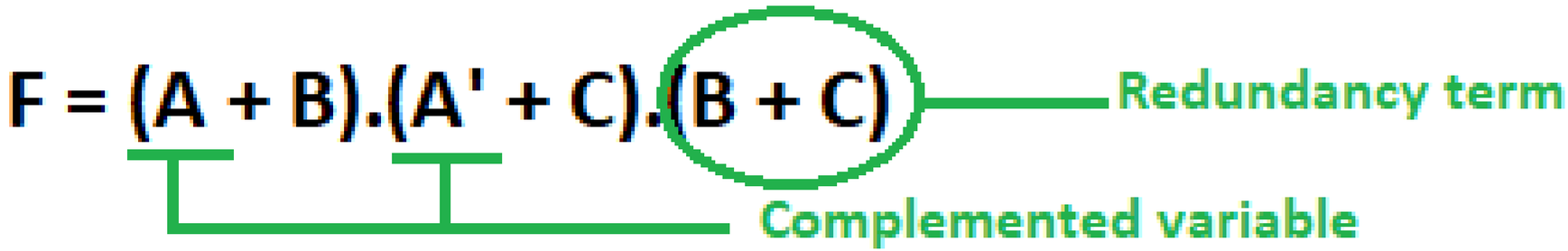
- **The equation must have three variables.**
- **Each variable is repeated twice.**
- **One variable must be present in the complemented form.**

RESULT OF Consensus theorem will be the expression including complemented term and uncomplemented term.



$$F = AB + BC' + AC = BC' + AC$$

- $F = (A + B).(A' + C).(B + C) = (A + B).(A' + C)$



BOOLEN EXPRESSION SIMPLIFICATION RULES AND EX

Name	AND form	OR form
Identity law	$1A = A$	$0 + A = A$
Null law	$0A = 0$	$1 + A = 1$
Idempotent law	$AA = A$	$A + A = A$
Inverse law	$A\bar{A} = 0$	$A + \bar{A} = 1$
Commutative law	$AB = BA$	$A + B = B + A$
Associative law	$(AB)C = A(BC)$	$(A + B) + C = A + (B + C)$
Distributive law	$A + BC = (A + B)(A + C)$	$A(B + C) = AB + AC$
Absorption law	$A(A + B) = A$	$A + AB = A$
De Morgan's law	$\overline{AB} = \bar{A} + \bar{B}$	$\overline{A + B} = \bar{A}\bar{B}$

- **reduce the given Boolean expression: F(X, Y, Z) = X′Y + YZ′ + YZ +XY′Z′**

  simplified form of the given Boolean expression is F(X, Y, Z) = Y+XZ′.

- **Reduce the following Boolean expression: F(P ,Q, R)=(P+Q)(P+R)**

Given, F(P ,Q, R)=(P+Q)(P+R) , Using distributive law, ⇒ F(P, Q, R) = P.P + P.R +Q.P + Q.R

Using Idempotent law, ⇒ F(P, Q, R) = P + P.R +Q.P + Q.R
Again using distributive law, we get ⇒ F(P, Q, R) = P(1+R) + Q.P + Q.R
Using dominance law, we can write ⇒ F(P, Q, R) = P + Q.P + Q.R
Again using distributive law, we get ⇒ F(P, Q, R) = (P+1).P+ Q.R
Therefore, using dominance law, we can get the reduced form as follows:⇒ F(P, Q, R) = 1.P+Q.R⇒ F(P, Q, R) = P+Q.R

Hence, the reduced form of F(P, Q, R) = (P+Q)(P+R) is F(P, Q, R) = P+Q.R.

What is the equivalent minimized expression for the Boolean expression x’y’z +yz+ xz?

solution : x’y’z +yz+ xz = z(x’y’+y+x)

= z(x’y’+x+y) = Z(x+y’+y) since x’y’+x = x+y’

= z(x+1) since y+y’ =1

= z

- **(AB’(C+BD) + A’B’)C minimize this equation.**

Solution: Given expression: (AB’(C+BD) + A’B’)C

Using distributive law, we can write;

(AB’(C+BD) + A’B’)C = (AB’C + AB’BD +A’B’) C Since, BB’ = 0

= (AB’C + 0 + A’B’)C = (AB’C + A’B’)C = AB’C + A’B’C

Now, take B’C as common,

= B’C (A + A’) = B’C (1) = B’C.

Hence, the reduced form of (AB’(C+BD) + A’B’)C is B’C.

- **Reduce the Boolean expression: A = XY + X(Y+Z) + Y(Y+Z)**

Solution: The given Boolean expression can be written as follows:

A = XY + XY + XZ + YY + YZ , Using Idempotent law, Y.Y = Y
Thus, A = XY + XY + XZ + Y + YZ , Again, XY + XY = XY, we get
A = XY + XZ + Y + YZ = XY + XZ + Y(1+Z)
Using null law, (i.e., 1+A = 1), we can write A = XY + XZ + Y.1
Using Identity law, 1.Y = Y. So, A = XY + XZ + Y
Now, the above form can be written as: A = Y(1+X) + XZ
A = Y.1 + XZ= Y + XZ

- minimize (A+B)(A+B’)(A’+C)

= (AA+AB’+AB+BB’)(A’+C) = (A+AB’+AB)(A’+C)

=(A+A(B+B’)(A’+C) = (A+A(1))(A’+C) = A(A’+C) = AA’+AC = AC

Practice Questions

Determine the simplified form of the Boolean expression xy + (~x)z + yz.

Find the complement of the Boolean expression: XY(Y’Z + XZ).

Find the reduced form of the Boolean expression: a’bc’ + a’bc + abc’ + abc.

Min terms and max terms

- We will get four Boolean product terms by combining two variables x and y with logical AND operation.
- These Boolean product terms are called as min terms or standard product terms. The min terms are x’y’, x’y, xy’ and xy.
- Similarly, we will get four Boolean sum terms by combining two variables x and y with logical OR operation.
- These Boolean sum terms are called as Max terms or standard sum terms. The Max terms are x + y, x + y’, x’ + y and x’ + y’.
- A=0, B=1, C=0 Maxterm is A+B'+C
- A=1, B=1, C=1 Maxterm is A'+B'+C'

- **While writing POS/ PRODUCT OF MAXTERMS , the following convention is to be followed:**

If variable A is Low(0) - A
A is High(1) - A'

- While writing SOP, the following convention is to be followed:

If variable A is Low(0) - A’
A is High(1) - A

	SOP	POS
1.	A way of representing boolean expressions as sum of product terms.	A way of representing boolean expressions as product of sum terms.
2.	SOP uses minterms. Minterm is product of boolean variables either in normal form or complemented form.	POS uses maxterms. Maxterm is sum of boolean variables either in normal form or complemented form.
3.	It is sum of minterms. Minterms are represented as ‘m’	It is product of maxterms. Maxterms are represented as ‘M’
4.	SOP is formed by considering all the minterms, whose output is HIGH(1)	POS is formed by considering all the maxterms, whose output is LOW(0)
5.	While writing minterms for SOP, input with value 1 is considered as the variable itself and input with value 0 is considered as complement of the input.	While writing maxterms for POS, input with value 1 is considered as the complement and input with value 0 is considered as the variable itself.

x	y	Min terms	Max terms
0	0	m0=x’y’	M0=x + y
0	1	m1=x’y	M1=x + y’
1	0	m2=xy’	M2=x’ + y
1	1	m3=xy	M3=x’ + y’

				Minterms		*Maxterms*
X	Y	Z		*Product Terms*		*Sum Terms*
0	0	0		$m_0 = \overline{X} \cdot \overline{Y} \cdot \overline{Z} = \min\left(\overline{X}, \overline{Y}, \overline{Z}\right)$		$M_0 = X + Y + Z = \max\left(X, Y, Z\right)$
0	0	1		$m_1 = \overline{X} \cdot \overline{Y} \cdot Z = \min\left(\overline{X}, \overline{Y}, Z\right)$		$M_1 = X + Y + \overline{Z} = \max\left(X, Y, \overline{Z}\right)$
0	1	0		$m_2 = \overline{X} \cdot Y \cdot \overline{Z} = \min\left(\overline{X}, Y, \overline{Z}\right)$		$M_2 = X + \overline{Y} + Z = \max\left(X, \overline{Y}, Z\right)$
0	1	1		$m_3 = \overline{X} \cdot Y \cdot Z = \min\left(\overline{X}, Y, Z\right)$		$M_3 = X + \overline{Y} + \overline{Z} = \max\left(X, \overline{Y}, \overline{Z}\right)$
1	0	0		$m_4 = X \cdot \overline{Y} \cdot \overline{Z} = \min\left(X, \overline{Y}, \overline{Z}\right)$		$M_4 = \overline{X} + Y + Z = \max\left(\overline{X}, Y, Z\right)$
1	0	1		$m_5 = X \cdot \overline{Y} \cdot Z = \min\left(X, \overline{Y}, Z\right)$		$M_5 = \overline{X} + Y + \overline{Z} = \max\left(\overline{X}, Y, \overline{Z}\right)$
1	1	0		$m_6 = X \cdot Y \cdot \overline{Z} = \min\left(X, Y, \overline{Z}\right)$		$M_6 = \overline{X} + \overline{Y} + Z = \max\left(\overline{X}, \overline{Y}, Z\right)$
1	1	1		$m_7 = X \cdot Y \cdot Z = \min\left(X, Y, Z\right)$		$M_7 = \overline{X} + \overline{Y} + \overline{Z} = \max\left(\overline{X}, \overline{Y}, \overline{Z}\right)$

Sum of minterms –

- The minterms whose sum defines the Boolean function are those which give the 1’s of the function in a truth table.
- Since the function can be either 1 or 0 for each minterm, and since there are 2^n minterms, one can calculate all the functions that can be formed with n variables to be (2^(2^n)).
- It is sometimes convenient to express a Boolean function in its sum of minterm form.

- Express the Boolean function F = A + B’C as standard sum of minterms.

A = A(B + B’) = AB + AB’
This function is still missing one variable, so
A = AB(C + C’) + AB'(C + C’) = ABC + ABC’+ AB’C + AB’C’
The second term B’C is missing one variable; hence,
B’C = B’C(A + A’) = AB’C + A’B’C
Combining all terms, we have
F = A + B’C = ABC + ABC’ + AB’C + AB’C’ + AB’C + A’B’C
But AB’C appears twice, and
according to theorem 1 (x + x = x), it is possible to remove one of those occurrences.
Rearranging the minterms in ascending order, we finally obtain
F = A’B’C + AB’C’ + AB’C + ABC’ + ABC
= m1 + m4 + m5 + m6 + m7
SOP is represented as Sigma(1, 4, 5, 6, 7)

Express the Boolean function F = xy + x’z as a product of maxterms

Solution –

F = xy + x’z = (xy + x’)(xy + z) = (x + x’)(y + x’)(x + z)(y + z) = (x’ + y)(x + z)(y + z)

x’ + y = x’ + y + zz’ = (x’+ y + z)(x’ + y + z’)

x + z = x + z + yy’ = (x + y + z)(x + y’ + z)

y + z = y + z + xx’ = (x + y + z)(x’ + y + z)

F = (x + y + z)(x + y’ + z)(x’ + y + z)(x’ + y + z’)

= M0*M2*M4*M5

POS is represented as Pi(0, 2, 4, 5)

- If the binary variable is ‘0’, then it is represented as complement of variable in min term and as the variable itself in Max term.
- Similarly, if the binary variable is ‘1’, then it is represented as complement of variable in Max term and as the variable itself in min term.
- min terms and Max terms are complement of each other. If there are ‘n’ Boolean variables, then there will be 2^n min terms and 2^n Max terms.

Canonical SoP and PoS forms

- we can express each output variable in following two ways.

Canonical SoP form
Canonical PoS form

Canonical SoP form

Canonical SoP form means Canonical Sum of Products form.

In this form, each product term contains all literals.

So, these product terms are nothing but the min terms. Hence, canonical SoP form is also called as sum of min terms form.

- First, identify the min terms for which, the output variable is one and then do the logical OR of those min terms in order to get the Boolean expression function corresponding to that output variable. This Boolean function will be in the form of sum of min terms.
-

Inputs			
p	q	r	f (OUTPUT)
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	1
1	1	1	1

- Boolean function of output is, f = p’qr + pq’r + pqr’ + pqr. This is the canonical SoP form of output,

f=m3+m5+m6+m7

f=∑m(3,5,6,7)

Canonical PoS form

- each sum term contains all literals.
- So, these sum terms are nothing but the Max terms.
- Hence, canonical PoS form is also called as product of Max terms form.
- First, identify the Max terms for which, the output variable is zero and then do the logical AND of those Max terms in order to get the Boolean expression function corresponding to that output variable.
- Consider the same truth table of previous example
  - the Boolean function of output is, f = p+q+r .p+q+r′ .p+q′+r .p′+q+r
  - This is the canonical PoS form of output, f.

f=M0.M1.M2.M4

f=∏M(0,1,2,4)

Standard SoP and PoS forms

- The main advantage of standard forms is that the number of inputs applied to logic gates can be minimized.
- In this form, each product term need not contain all literals.
- Standard SoP form of output variable in two steps.

1. Get the canonical SoP form of output variable
2. Simplify the above Boolean function, which is in canonical SoP form.

- Convert the following Boolean function into Standard SoP form.

f = p’qr + pq’r + pqr’ + pqr

f = pq + qr + pr (standard)