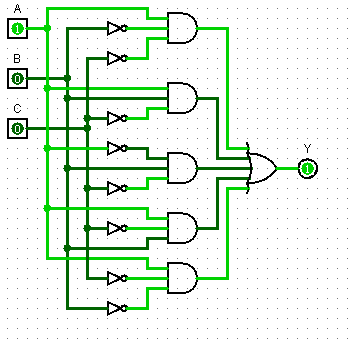
**Lab 3**

**By Rushan khan**

**Submitted to Sir Rafay Shaikh**

**Q:1 Simplify given expression using Standard Sum of Product, also show step by step process of building a circuit and designing a truth table.**

**Q1:**

**1: F1 (A, B, C) = A B’ C’ + B C’ + A C’**

Sol:

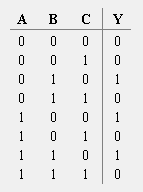
Multiply B ~C with (A+A’) and A C’ with (B+B’)

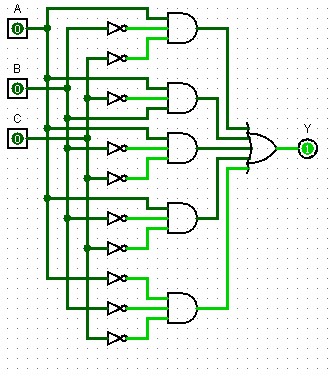
= A B’ C’ + B C’ (A+A’) +A C’(B+B’)

= A B’ C’ + B C’ A + B C’ A’ + A C’ B + A C’ B’

= A B’ C’ + A B C’+ A’ B C’+ A C’ B + A C’ B’

Ans:



**2: F2 (A, B, C) = A~B~C + A ~C + ~B ~C**

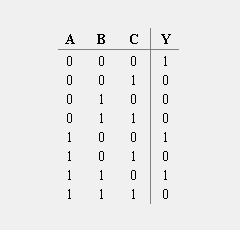
Sol:

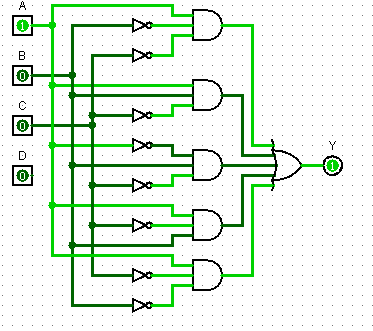
=A~B~C + A~C(B+~B) +~B~C(A+~A)

=A~B~C+A~CB+A~C~B+~B~CA+~B~C~A

=A~B~C + A~CB + A~B~C + A~B~C + ~A~B~C

Ans



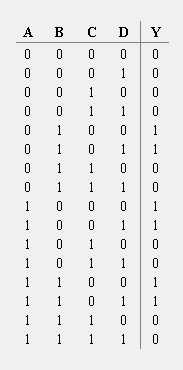
**3: F1 (A, B, C, D) = A’B’C’D’ + ABC’ + AC’**

SOL:

=A’B’C’D’ + ABC’(D+D’) + AC’(B+B’) [D+D’]

=A’B’C’D’ + ABC’D + ABC’D’ + ABC’D +AB’C’D’

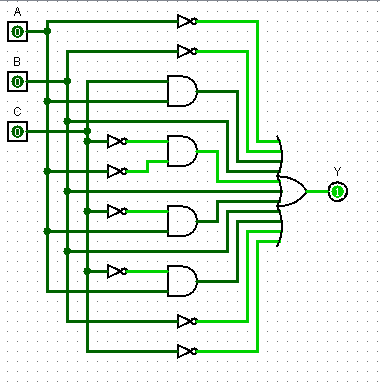
ANS



Q;2 Simplify given expression using Standard Product of Sum, also show step by step process of building a circuit and designing a truth table

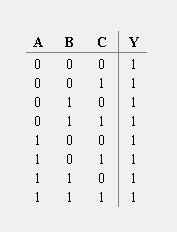
**1: F1 (A,B,C)= (A’+B’+C)(B+C’)(A+C’)**

SOL:

= (A’+B’+C)(B+C’)(A+A’)(A+C’)(B+B’)

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

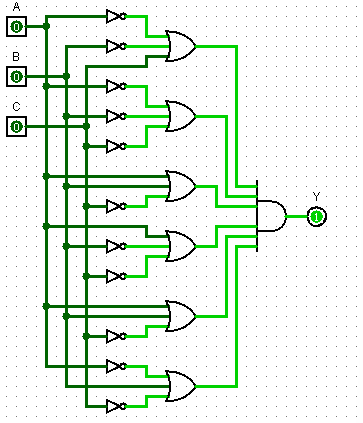
ANS

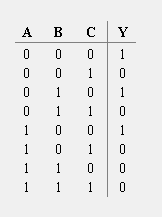


**2: F (A,B,C) = (A’ + B’) (A + C’) (B + C’)**

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

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



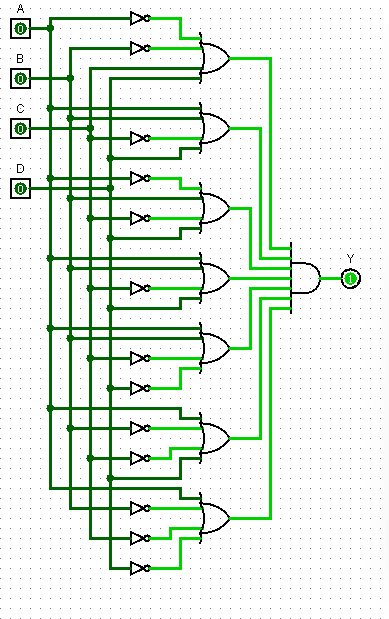


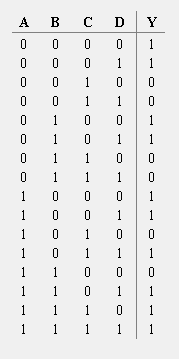
**3: F (A,B,C,D) = (~A+~B+C+D )( B+~C+D) ( A+~C)**

= (~A+~B+C+D ) ( B+~C+D) (A + ~A) ( A+~C) (B + ~B)

= (~A+~B+C+D ) (A + B+~C+D) (~A + B+~C+D) ( A+B+~C) (D + ~D) ( A+~B+~C) (D + ~D)

= (~A+~B+C+D ) (A + B+~C+D) (~A + B+~C+D) ( A+B+~C +D) ( A+B+~C + ~D) ( A+~B+~C +D) ( A+~B+~C + ~D)





**Q:3 Why do we convert SOF & POS into their Canonical form?**

SOF (Sum of Products) and POS (Product of Sums) are two standard forms used in Boolean algebra to represent Boolean functions. The canonical form of SOF and POS are the simplest and most standard forms of representation for Boolean functions.

Converting a Boolean function to its canonical form involves expressing the function as a combination of fundamental Boolean operations, such as AND, OR, and NOT gates, in a standard way. The resulting canonical form is unique, which means that there is only one canonical form for a given Boolean function.

There are several reasons why we convert SOF and POS into their canonical form, including:

Simplification: The canonical form of a Boolean function is the simplest form of representation, which means that it is easier to understand and manipulate. This can be helpful when trying to simplify complex Boolean functions.

Optimization: Converting SOF and POS into their canonical form can help to optimize digital circuits, making them faster and more efficient.

Standardization: The canonical form of SOF and POS provides a standard way of representing Boolean functions, which makes it easier for designers and engineers to communicate and collaborate on projects.

Minimization: The canonical form of a Boolean function is the minimal representation of that function, which means that it requires the fewest possible gates to implement. This can help to reduce the cost and complexity of digital circuits

**Q:4 What is Combinational Analysis?**

Combinational analysis is an important tool in digital logic design and is used in a wide range of applications, including computer hardware design, telecommunications, and control systems. It is particularly useful in designing circuits that perform arithmetic and logic operations, such as adders, subtractors, and comparators.

**Q:5 What are Minterms and Maxterms?**

What are minterms and Maxterms?

In Boolean algebra, minterms and maxterms are two standard terms used to represent Boolean functions.

A minterm is a product term that includes all the input variables of a Boolean function, where each variable appears either in its true or complemented form. For a Boolean function with n input variables, there are 2^n possible minterms, with each minterm corresponding to a unique combination of input variables.

For example, consider a Boolean function with three input variables A, B, and C. The minterm for the input combination A=1, B=0, C=1 is represented as A'BC.

A maxterm, on the other hand, is a sum term that includes all the input variables of a Boolean function, where each variable appears either in its true or complemented form. For a Boolean function with n input variables, there are also 2^n possible maxterms, with each maxterm corresponding to a unique combination of input variables.

For example, consider a Boolean function with three input variables A, B, and C. The maxterm for the input combination A=0, B=1, C=0 is represented as A+B'+C'.

The minterm and maxterm representations of a Boolean function can be used to simplify the function and implement it using logic gates. The minterm representation is useful for implementing the function using a series of AND gates, while the maxterm representation is useful for implementing the function using a series of OR gates.