SNS COLLEGE OF ENGINEERING

(Autonomous)

DEPARTMENT OF CSE - IoT

COURSE NAME:19EC306 / DIGITAL CIRCUITS

II YEAR/III SEMESTER

UNIT:1- MINIMIZATION TECHNIQUES AND LOGIC GATES

TOPIC: MINIMIZATION OF BOOLEAN EXPRESSIONS, MIN TERMS, MAX TERMS

Outline

- In mathematics, expressions are simplified for a number of reasons, for instance simpler expression are easier to understand and easier to write down, they are also less prone to error in interpretation but, most importantly, simplified expressions are usually more efficient and effective when implemented in practice.
- A Boolean expression is composed of variables and terms. The simplification of Boolean expressions can lead to more effective computer programs, algorithms and circuits.

Methods of minimization

- Minimisation can be achieved by a number of methods, three well known methods are:
  1. Algebraic Manipulation of Boolean Expressions
  2. Tabular Method of Minimization
  3. Karnaugh Maps

Algebric manipulation of Boolean expressions

$x'y' + xyz + x'y$

$= x'(y' + y) + xyz$ [Distributive: $x'y' + x'y = x'(y' + y)$]

$= x' \bullet 1 + xyz$ [complement: $x' + x = 1$]

$= x' + xyz$ [identity: $x' \bullet 1 = x'$]

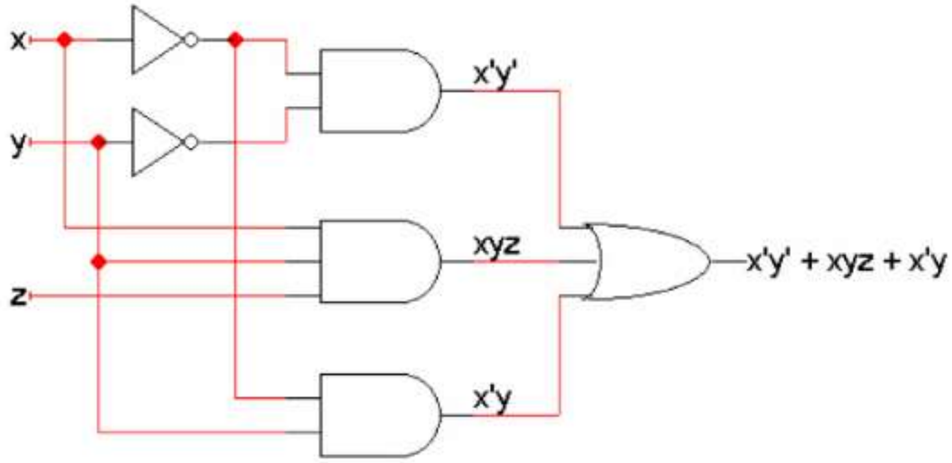
$= (x' + x)(x' + yz)$ [Distributive]

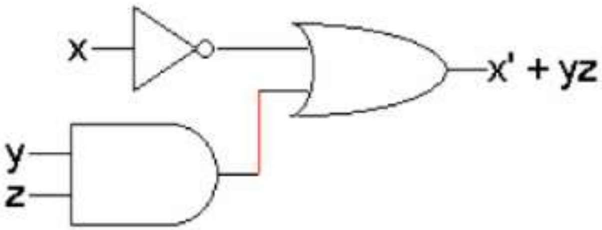
$= 1 \bullet (x' + yz)$ [complement: $x' + x = 1$]

$= x' + yz$ [identity]

Circuit reduction

- Here are two different but *equivalent* circuits.
- In general the one with fewer gates is "better":
  - It costs less to build
  - It requires less power
  - But we had to do some work to find the second form





Representation of a function

- A function can be specified or represented in any of the following ways:
  - A truth table
  - A circuit
  - A Boolean expression
    - SOP (Sum Of Products)
    - POS (Product of Sums)
    - Canonical SOP
    - Canonical POS

MINTERMS

Minterms:

- A form is canonical, if representation of a function in this form is unique
  - Truth table is canonical representation
  - Uses minterms as basic components.

x	y	minterm	designation
0	0	x'y'	m_0
0	1	x'y	m_1
1	0	xy'	m_2
1	1	xy	m_3

x	y	z	minterm	designation
0	0	0	x'y'z'	m_0
0	0	1	x'y'z	m_1
0	1	0	x'yz'	m_2
0	1	1	x'yz	m_3
1	0	0	xy'z'	m_4
1	0	1	xy'z	m_5
1	1	0	xyz'	m_6
1	1	1	xyz	m_7

- Each minterm is obtained from an AND term of the n variables, with each variable being primed if the corresponding bit of the binary number is a 0 and unprimed if a 1
- A symbol for each minterm is also shown in the table and is of the form m_j, where the subscript j denotes the decimal equivalent of the binary number of the minterm designated.

MAXTERMS

Maxterms:

- In a similar fashion, n variables forming an OR term, with each variable being primed or unprimed, provide 2^n possible combinations, called maxterms, or standard sums

x	y	minterm	designation
0	0	x+y	M_0
0	1	x+y'	M_1
1	0	x'+y	M_2
1	1	x'+y'	M_3

			Maxterms	
x	y	z	Term	Designation
0	0	0	$x + y + z$	M_0
0	0	1	$x + y + z'$	M_1
0	1	0	$x + y' + z$	M_2
0	1	1	$x + y' + z'$	M_3
1	0	0	$x' + y + z$	M_4
1	0	1	$x' + y + z'$	M_5
1	1	0	$x' + y' + z$	M_6
1	1	1	$x' + y' + z'$	M_7

- It is important to note that
  - Each maxterms is obtained from an OR term of the n variables, with each variable being unprimed if the corresponding bit is a 0 and primed if a 1.
  - Each maxterms is the complement of its corresponding minterm and vice versa

MAXTERMS

Maxterms:

- In a similar fashion, n variables forming an OR term, with each variable being primed or unprimed, provide 2^n possible combinations, called maxterms, or standard sums

x	y	minterm	designation
0	0	x+y	M_0
0	1	x+y'	M_1
1	0	x'+y	M_2
1	1	x'+y'	M_3

			Maxterms	
x	*y*	*z*	Term	Designation
0	0	0	$x + y + z$	M_0
0	0	1	$x + y + z'$	M_1
0	1	0	$x + y' + z$	M_2
0	1	1	$x + y' + z'$	M_3
1	0	0	$x' + y + z$	M_4
1	0	1	$x' + y + z'$	M_5
1	1	0	$x' + y' + z$	M_6
1	1	1	$x' + y' + z'$	M_7

- It is important to note that
  - Each maxterms is obtained from an OR term of the n variables, with each variable being unprimed if the corresponding bit is a 0 and primed if a 1.
  - Each maxterms is the complement of its corresponding minterm and vice versa

SUM OF MINTERMS

Express the Boolean function $F=A+B'C$ as a sum of minterms. The function has three variables: A,B, and C. The first term A is missing two variables; therefore,

$$A=A(B+B')=AB+AB'$$

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

$$F=A+B'C$$
$$=ABC+ABC'+\underline{AB'C}+AB'C'+\underline{AB'C}+A'B'C$$

But AB'C appears twice, and according theorem 1 (x+x=x), it is possible to remove one of those occurrences. Rearranging the minterms in ascending order, we finally obtain

$$F=A'B'C+AB'C'+AB'C+ABC'+ABC=m_1+m_4+m_5+m_6+m_7$$

$$F(A,B,C)=\sum(1,4,5,6,7)$$

The summation symbol $\sum$ stands for the ORing of terms; the numbers following it are the indices of the minterms of the function

PRODUCT OF MAX TERMS

To express a Boolean function as a product of maxterms, it must first be brought into a form of OR terms. This may be done by using the distributive law, $x + yz = (x + y)(x + z)$. Then any missing variable x in each OR term is ORed with xx'. The procedure is clarified in the following example.

Example: Express the Boolean function $F=xy + x'z$ as a product of maxterms. First, convert the function into OR terms by using the distributive law:

$$F=xy + x'z=(xy+x')(xy+z)$$
$$=(x+x')(y+x')(x+z)(y+z)$$
$$=(x'+y)(x+z)(y+z)$$

The function has three variables:x,y, and z. Each OR term is missing one variable therefore,

$$x'+y=x'+y+zz'=(x'+y+z)(x'+y+z')$$
$$x+z=x+z+yy'=(x+y+z)(x+y'+z)$$
$$y+z=y+z+xx'=(x+y+z)(x'+y+z)$$

Combining all the terms and removing those which appear more than once, we finally obtain

$$F=(x+y+z)(x+y'+z)(x'+y+z)(x'+y+z')$$
$$=M_0M_2M_4M_5$$

A convenient way to Express this function is : $F(x, y, z) = \Pi(0, 2, 4, 5)$

Thank you