Chapter Two (con’t)

Where were we?

- Algebraic manipulation can optimize functions
  - Minimizes the number of literals
    - Literal
      - A single variable within a term; whether complemented or not.
  - Minimization of the number of literals and/or terms often leads to a simpler circuit

Complement of a Function

- Derived algebraically via the application of DeMorgan's Theorem
  - Shown in Table 2.1 for 2 variables
  - Can be extended to 3 or more

Proof of 3 Variable DeMorgan's

Given the function $F = A + B + C$

Then $F' = (A + B + C)'$

$= (A + x)'$	Let $x = B + C$
$= A'x'$	2 variable DeMorgan's
$= A'(B + C)'$	Back-substitution
$= A'(B'C')$	2 variable DeMorgan's
$= A'B'C'$	Associativity

Example

$F = x'yz' + x'y'z$

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

$= (x'yz')'(x'y'z)'$

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

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

Another Way

- Use Duality to your advantage

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

$= x + (y' + z')(y + z)$ The dual of F

$= x' + (y + z)(y' + z')$ Complement each literal

$= x' + yy' + zy' + z'y + zz'$

$= x' + y'z + yz'$

Algebra Reminder

- Sum
  - the result of addition
  - 1 + 2 = 3, 3 is the sum
- Product
  - The result of multiplication
  - 1 * 2 = 2, 3 is the product

Some symbology

- Each of these (sum and product) have a symbol
  - $\sum$ (Sigma)
  - $\prod$ (Pi)

Definition time again

Minterms and Maxterms…

Sum of Products and Products of Sums…

Minterms

- Consider 2 variable (x, y) and AND them.
  - Result: x’y’, x’y, xy’, xy
  - Each of which is a <u>minterm</u>
  - AKA <u>standard product</u>

x	y	
0	0	x’y’
0	1	x’y
1	0	xy’
1	1	xy

Table 2.3

			Minterms	
x	y	z	Term	Designation
0	0	0	$x'y'z'$	m_0
0	0	1	$x'y'z$	m_1
0	1	0	$x'yz'$	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

Designation of the minterm is in the form of m_j where j = the decimal equivalet of the binary number of the row

Maxterms

- Similar to minterms,
  - Still 2 variables
  - Except forming an OR term
  - AKA <u>standard sum</u>

x	y	
0	0	x + y
0	1	x + y’
1	0	x’ + y
1	1	x’ + y’

Table 2.3

Table 2.3
Minterms and Maxterms for Three Binary Variables

			Minterms		Maxterms	
x	**y**	**z**	**Term**	**Designation**	**Term**	**Designation**
0	0	0	$x'y'z'$	m_0	$x + y + z$	M_0
0	0	1	$x'y'z$	m_1	$x + y + z'$	M_1
0	1	0	$x'yz'$	m_2	$x + y' + z$	M_2
0	1	1	$x'yz$	m_3	$x + y' + z'$	M_3
1	0	0	$xy'z'$	m_4	$x' + y + z$	M_4
1	0	1	$xy'z$	m_5	$x' + y + z'$	M_5
1	1	0	xyz'	m_6	$x' + y' + z$	M_6
1	1	1	xyz	m_7	$x' + y' + z'$	M_7

Designation of the maxterm is M_j

Example

$F = x'y'z + xy'z' + xyz$

Minterms: $m_1 + m_4 + m_7$

$F' = x'y'z' + x'yz' + x'yz + xy'z$

$+ xyz'$

x	**y**	**z**	**F**
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

continued

$F' = x'y'z' + x'yz' + x'yz + xy'z + xyz'$

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

$F = (x + y + z)(x + y' + z)(x + y' + z')(x' + y + z')$
$(x' + y' + z)$
$= M_0M_2M_3M_5M_7$

- NOTE: Mistake on pg 52. This term is missing.

General Case

- n variables
- Each minterm is the AND of the inputs
  - Each variable is primed iff the corresponding input is 0.
- Each Maxterm is the Or of the inputs
  - Each variable is primed iff the corresponding input is 1.
- Can still be read from a Truth table

Canonical Forms

- A function expressed as a
  - Sum of minterms, or a
  - Product of Maxterms

Sum of Minterms

- Any function can be expressed in a sum of minterms format.
  - May have to manipulate the function to do so.
  - Each term must include all possible variables (literals)
- Remember each minterm must include all literals (a literal may be complemented)

Example

$F = A + B'C$

First term A is missing 2 variables

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

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

$A = ABC + ABC' + AB'C + AB'C'$

Second term B'C is missing 1 variable

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

- Put it all together

$F = A + B'C$

$= ABC + ABC' + AB'C + AB'C' + AB'C + A'B'C$

$= ABC + ABC' + AB'C + AB'C' + A'B'C$

$= A'B'C + AB'C' + AB'C + ABC' + ABC$ Reordered

- Minterms

$F = m_1 + m_4 + m_5 + m_6 + m_7$

Much easier way

- Construct a truth table for the function
- Derive the minterms directly from the truth table.

Example

Truth Table for $F = A + B'C$

A	*B*	*C*	*F*
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

From the truth table, we can see what the minterms are.

Any row where the Function = 1, is a minterm.

Product of Maxterms

- A boolean function can also be expressed as a product of maxterms.
- This too can be derived from an equation.
- But it is easier to derive it from the truth table.

Example

Truth Table for F = A + B′C

A	*B*	*C*	*F*
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Any row where the Function = 0, is a Maxterm.

Remember a Maxterm is a sum.

Ex $M_0 = A + B + C$

Examples

- $F(A, B, C) = \sum(1, 4, 5) = m_1 + m_4 + m_5$
  - The sum of m_1, m_4, m_5
- $F'(A, B, C) = \sum(0, 2, 3, 6, 7)$

  $= m_0 + m_2 + m_3 + m_6 + m_7$
- Complement F'

  $F'' = F = \prod(0, 2, 3, 6, 7) = M_0M_2M_3M_6M_7$

Things to Note

- The canonical forms are easily read from a truth table
- They are basic forms
- They are seldom in an optimized form
  - Why?
- Sum of minterms AKA Canonical Sum of Products
- Product of Maxterms AKA Canonical Product of Sums

Standard Forms

- Sum of Products
  - A Boolean expression containing AND terms (products) which are OR'd together (summed).

    $$F = x + xy + x'yz'$$

- Logic diagram will consist of AND gates and an OR gate.

Standard Forms

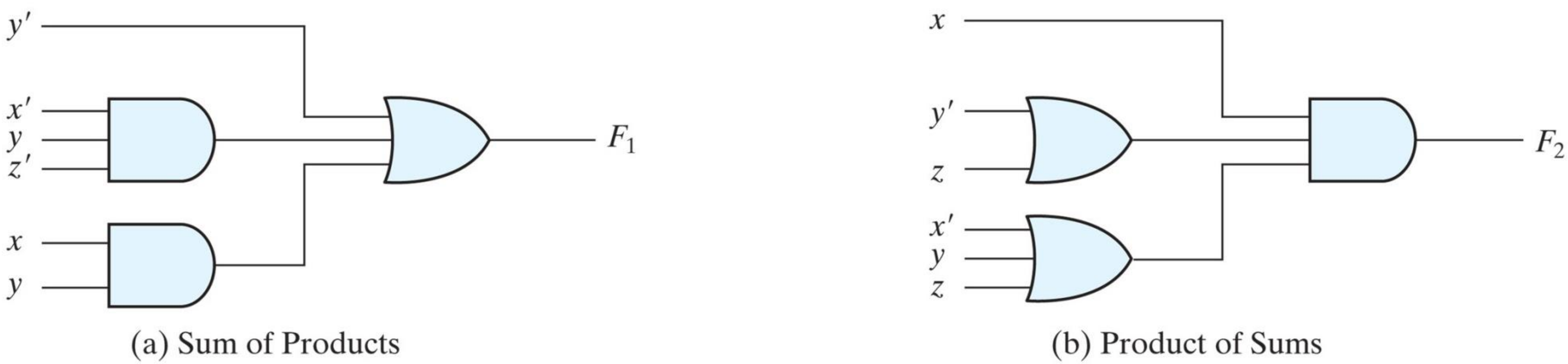
- Product of Sums
  - A Boolean expression containing OR terms (sums) which are AND’d together (product).

$$F = y\,(x + x'z')$$

- Logic diagram will consist of OR gates and an AND gate.

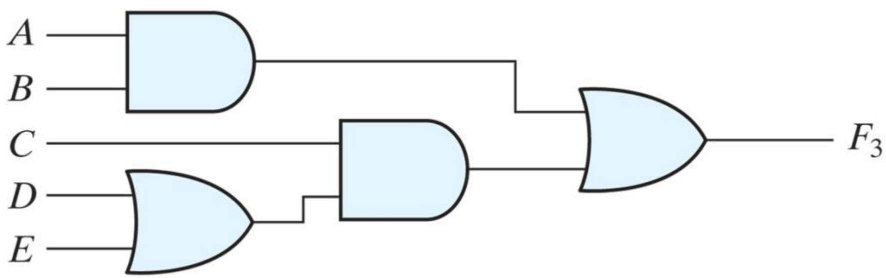
Standard Forms

- Always result in a two-level implementation.

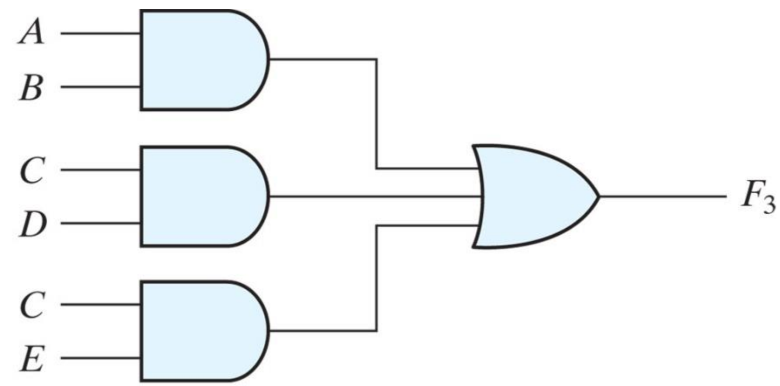


Standard Form

- In general, a two-level implementation is preferred.
  - Functions can be implemented in other forms, but will inherently have more delay.
- Delay is caused by the components which create each gate and is a fundamental concern to designers.



(a) $AB + C(D + E)$



(b) $AB + CD + CE$

Word of Caution

- Sum of minterms
  - Also known as the Canonical Sum of Products
- Product of Maxterms
  - Also known as the Canonical Product of Sums
- But where the standard forms may be optimized, the canonical forms are not likely to be.

$$F = x + y'z$$

Digital Logic Operations

- AND, OR, NOT form the basis of all operations.
- NAND/NOR
  - Examined in Lab 1
  - Outputs are negated AND/OR respectively

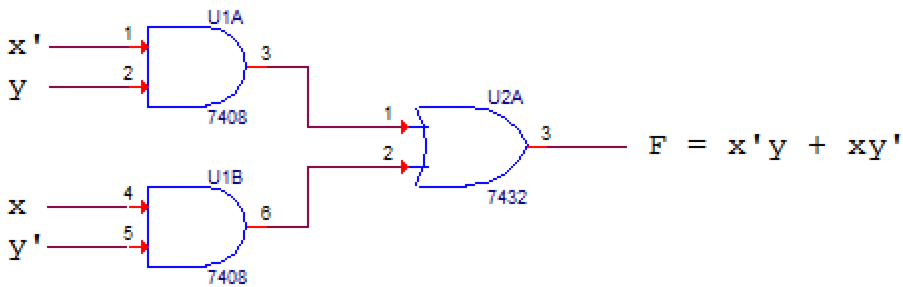
Application of Concepts

Problem Statement: You want to write an equation which checks if the two inputs are different (i.e. $x \neq y$)

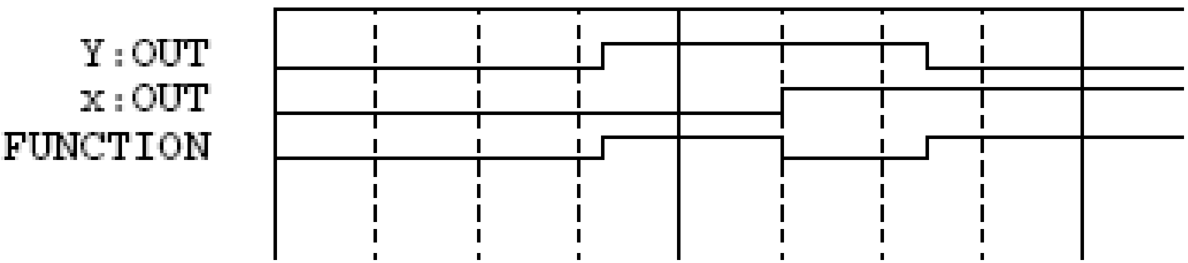
- Good place to start is with a truth table.

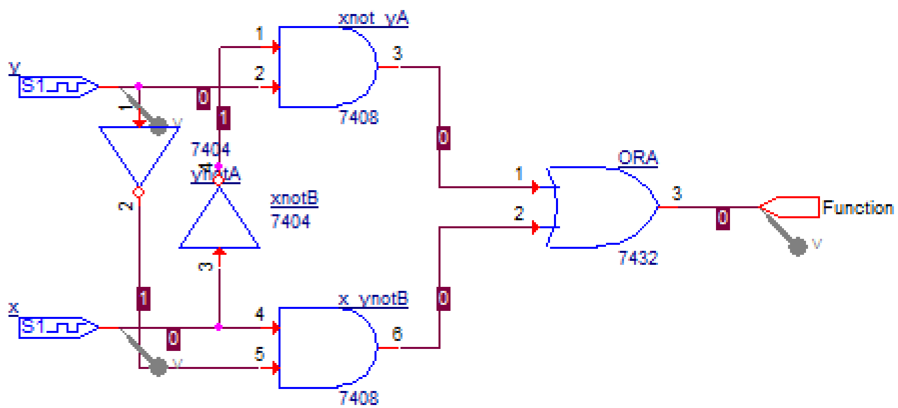
$x \neq y$

$F = x'y + xy'$



x	y	F





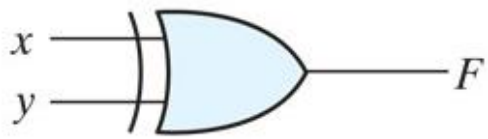
Desired Result

x	y	F
0	0	0
0	1	1
1	0	1
1	1	0

XOR

x or y but not both

Exclusive-OR
(XOR)



$$F = xy' + x'y$$
$$= x \oplus y$$

x	y	F
0	0	0
0	1	1
1	0	1
1	1	0