



University of  
Massachusetts  
Amherst

## Engin112 – Lectures 6-8

### Binary Logic and Boolean Algebra

Maciej Ciesielski  
Department of Electrical and Computer Engineering  
09/19-23/2011

## Recap from last lecture

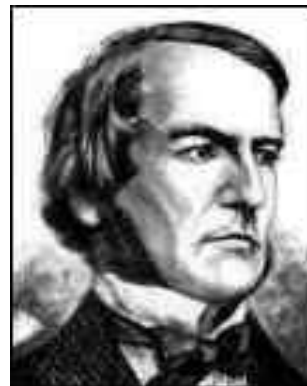
- **Codes**
  - Assignment of bit sequence to represent discrete elements
  - BCD code and addition
  - Error detecting codes (parity check)
  
- **Today's lecture**
  - Binary logic
  - Boolean algebra

# Binary Logic

- Binary logic uses two possible values
  - '1' and '0'
  - 'yes' and 'no'
  - 'true' and 'false'
  - Can be represented by variables:  $A, B, C, x, y, z, \dots$
- Logic functions modify input values
  - What is the minimum number of inputs?
    - » 1 input -> pretty uninteresting, inverter or wire only
    - » 2 inputs -> basic logic functions
- What are possible logic functions?
  - NOT, AND, OR
  - Others can be derived from those (NAND, XOR, etc.)

# George Boole

- Father of Boolean algebra
- Boole came up with a type of linguistic algebra, the three most basic operations of which were (and still are) AND, OR and NOT. It was these three functions that formed the basis of his premise, and were the only operations necessary to perform comparisons or basic mathematical functions.
- Boole's system (detailed in his 'An Investigation of the Laws of Thought, on Which Are Founded the Mathematical Theories of Logic and Probabilities', 1854) was based on a binary approach, processing only two objects - the yes-no, true-false, on-off, zero-one approach.
- Surprisingly, given his standing in the academic community, Boole's idea was either criticized or completely ignored by the majority of his peers.
- Eventually, one bright student, **Claude Shannon** (1916-2001), picked up the idea and ran with it



George Boole (1815 - 1864)

# NOT Function

- Complement operation (inversion)

- Single input
- Inverts value of input
- Symbolized by prime  $x'$  or overbar  $\bar{x}$

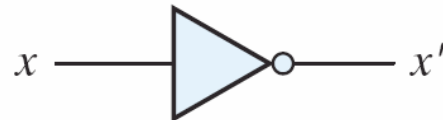
- Truth table

- Input combinations on the left
- Output of function on the right

input	output
$x$	$x'$
0	1
1	0

- Graphic symbol

- NOT gate (inverter)
- Little circle indicates inversion



# AND Function

- Operation to check if two conditions are met

- Two inputs
- Output = 1 if and only if both inputs are 1
- Symbolized by dot or absence of operator

$x \cdot y$  (also written as  $xy$ )

- Truth table

- Needs to consider  $2^2 = 4$  input combinations

inputs	output
$x$ $y$	$z$
0 0	0
0 1	0
1 0	0
1 1	1

- Graphic symbol

- AND gate



# OR Function

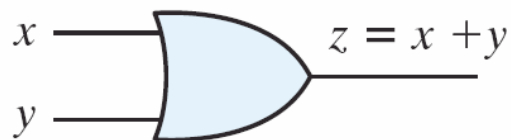
- Operation to check if at least one condition is met
  - Two inputs
  - Output = 1 if any one or both inputs are 1
  - Symbolized by “plus” sign:  $x + y$

- Truth table

x	y	z
0	0	0
0	1	1
1	0	1
1	1	1

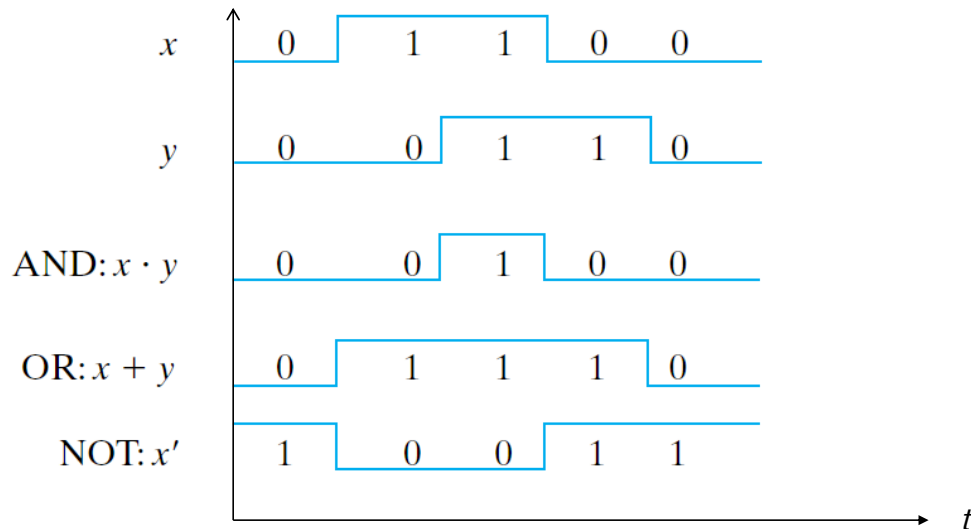
- Graphic symbol

- OR gate



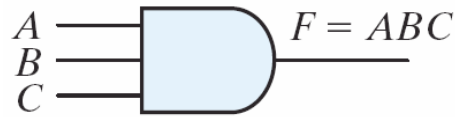
# Evaluation of Logic Functions

- Timing diagrams (waveforms)
  - Horizontal axis is time ( $t$ )
  - Vertical axis shows signals, each with two different voltage levels

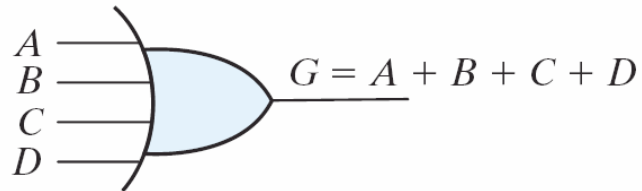


## Multiple Inputs

- Two inputs might not be enough
- 3-input AND gate:



- 4-input OR gate:



- What are the rules for aggregating functions?
  - Boolean algebra

## Reminder

- Homework #1 due Wednesday BEFORE class !
- Make sure to provide on the first/cover page the following:
  - » ENGIN 112 Fall 2011
  - » Homework Number, date submitted
  - » Your full Name (no ID needed)
  - » Your Discussion Section number
- Please paginate + staple your homework !

# Algebras

- **What is an algebra?**
  - Mathematical system consisting of
    - » Set of elements
    - » Set of operators
    - » Axioms or postulates
- **Why is it important?**
  - Defines rules of “calculations”
- **Example: arithmetic on natural numbers**
  - Set of elements:  $N = \{1,2,3,4,\dots\}$
  - Operator:  $+$ ,  $-$ ,  $\bullet$  (or  $\times$ )
  - Axioms: associativity, distributivity, closure, identity elements, etc.
- **Note: operators with two inputs are called binary**
  - Relate to two objects
  - Does not mean they are restricted to binary numbers!
- **Operator(s) with one input are called unary**

# Axioms of Algebraic Structures

- **Common axioms (or postulates):**
  - 1. Closure:**
    - » A set  $S$  is closed with respect to a binary operator  $*$  if the operator specifies a rule for obtaining an element of  $S$ .
    - » Example:  $S = N$  and  $* = +, \bullet$  (not  $-$ )
  - 2. Associativity:**
    - » A binary operator  $*$  on a set  $S$  is said to be associative if  $(x * y) * z = x * (y * z)$
    - » Example:  $S = N$  and  $* = +, \bullet$  (not  $-$ )
  - 3. Commutativity:**
    - » A binary operator  $*$  on a set  $S$  is said to be commutative if  $x * y = y * x$
    - » Example:  $S = N$  and  $* = +, \bullet$  (not  $-$ )

# Axioms of Algebraic Structures

## 4. Identity element:

- » Set  $S$  is said to have an identity element with respect to binary operation  $*$  on  $S$  if there exists an element  $e \in S$  such that

$$e * x = x * e = x \quad \text{for every } x \in S$$

- » Example:  $S = \mathbb{Z}$ ,  $* = +$ ,  $e = 0$ ; or  $* = \cdot$  ( $\times$ ),  $e = 1$

## 5. Inverse:

- » A set  $S$  having the identity element  $e$  with respect to a binary operator  $*$  is said to have an inverse whenever, for every  $x \in S$ , there exists an element  $y \in S$  such that  $x * y = e$

- » Example:  $S = \mathbb{Z}$ ,  $* = +$ ,  $e = 0$ ,  $y = -x$

## 6. Distributivity:

- » If  $*$  and  $\#$  are two binary operators on a set  $S$ ,  $*$  is said to be distributive over  $\#$  whenever  $x * (y \# z) = (x * y) \# (x * z)$

- » Example:  $S = \mathbb{N}$ ,  $* = \cdot$ ,  $\# = +$

# Example Algebra: Field

- Instance of an algebraic structure is a *field*
  - A field consists of
    - » Set of elements
    - » Two binary operators (each having properties 1–5)
    - » Both operators combined have property 6 (distributivity)
- Example: field for ordinary algebra on real numbers
  - Binary operator  $+$  defines addition
  - Additive identity is 0
  - Additive inverse defines subtraction
  - Binary operator  $\cdot$  defines multiplication
  - Multiplicative identity is 1
  - Multiplicative inverse defines division
  - Distributive law:  $\cdot$  over  $+$  :  $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$

# Boolean Algebra

- We need to define algebra for binary values
  - Developed by George Boole in 1850's
  - Algebraic structure with set of elements  $B$  with two operators  $(+, \cdot)$  satisfying postulates
- Huntington postulates for Boolean algebra (1904):



Edward V. Huntington (1874 – 1952, USA)

- Math instructor at Williams College and professor at Harvard.
- Devised sets of axioms for many mathematical systems: a group, an abelian group, a Boolean algebra, geometry, the real number field, and the complex numbers.
- He showed that a Boolean algebra could be defined in terms of a single binary and a single unary operation. This is known as Huntington's theorem (1933).

# Huntington Postulates

- Huntington postulates for Boolean algebra (1904):
  1. Closure with respect to operator  $+$  and operator  $\cdot$
  2. Identity element  $0$  for operator  $+$  and  $1$  for operator  $\cdot$  ( $e * x = x$ )
  3. Commutativity with respect to  $+$  and  $\cdot$   
 $x+y = y+x$ ,  $x \cdot y = y \cdot x$
  4. Distributivity of  $\cdot$  over  $+$ , and  $+$  over  $\cdot$   
 $x \cdot (y+z) = (x \cdot y) + (x \cdot z)$  and  $x + (y \cdot z) = (x+y) \cdot (x+z)$
  5. Complement for every element  $x$  is  $x'$  with  $x+x'=1$ ,  $x \cdot x'=0$
  6. There are at least two elements  $x, y \in B$  such that  $x \neq y$
- Note:  $+$  and  $\cdot$  are chosen to be intuitive
  - Do not blindly substitute with ordinary algebra, it won't work !

# Boolean vs. Ordinary Algebra

## Differences

- Postulates do not include associativity
  - » Can be derived for both operators
- Distributivity of + over • holds for Boolean, but not for ordinary algebra
  - »  $x+(y \cdot z) = (x+y) \cdot (x+z)$
- Boolean algebra does not have inverse elements for + or •
  - » Thus, no subtraction or division operators
- Complement is not defined in ordinary algebra
- Set of elements in Boolean algebra not yet defined
  - » But there must be at least two elements

## Questions

- What is the set of elements?
- What are the rules of operation for + and • ?

# Two-valued Boolean Algebra

## Two-valued Boolean is defined as

- A set of elements  $B = \{0,1\}$

“+” operator:

x	y	$x+y$
0	0	0
0	1	1
1	0	1
1	1	1

“•” operator:

x	y	$x \cdot y$
0	0	0
0	1	0
1	0	0
1	1	1

complement ‘

x	$x'$
0	1
1	0

## Observation

- + is OR, • is AND, ' is NOT
- Other notation:  $+ = \vee$ ,  $\bullet = \wedge$ , NOT =  $\neg$

# Check Huntington Postulates

▪ Verify that postulates hold for Boolean algebra

1. Closure?
  - » Yes, all operations with  $\{0,1\} \in B$
2. Identity?
  - » Yes, 0 for + and 1 for •
3. Commutativity?
  - » Yes, for +:  $0+0=0, 1+1=1, 0+1=1+0=1$
  - » Yes, for •:  $0 \cdot 0=0, 1 \cdot 1=1, 0 \cdot 1=1 \cdot 0=0$
4. Distributivity?
  - » Yes, use truth table (p. 40 in Mano) for • over + and + over •
5. Complement?
  - » Yes,  $x+x'=1$ :  $0+0'=0+1=1$  and  $1+1'=1+0=1$
  - » Yes,  $x \cdot x'=0$ :  $0 \cdot 0'=0 \cdot 1=0$  and  $1 \cdot 1'=1 \cdot 0=0$
6. Two distinct values?
  - » Yes,  $0 \neq 1$

Huntington postulates:

- Post. 1:** closure  
**Post. 2:** (a)  $x+0=x$ , (b)  $x \cdot 1=x$   
**Post. 3:** (a)  $x+y=y+x$ , (b)  $x \cdot y=y \cdot x$   
**Post. 4:** (a)  $x(y+z) = xy+xz$ ,  
 (b)  $x+yz = (x+y)(x+z)$   
**Post. 5:** (a)  $x+x'=1$ , (b)  $x \cdot x'=0$

x	y	x+y	x	y	x • y
0	0	0	0	0	0
0	1	1	0	1	0
1	0	1	1	0	0
1	1	1	1	1	1

# Boolean Algebra

- Two-value Boolean algebra satisfies Huntington postulates
  - We can use binary logic and consider it an algebra
  
- Next:
  - Theorems that can be derived from postulates
  - Canonical forms (uniqueness of representation)
  - Duality of Boolean algebra
  - Boolean functions

# Recap from Last Lecture

- Boolean algebra
  - Algebra axioms, postulates
  - Huntington's postulates
- Boolean functions
  - Algebraic expression
  
- Today's lecture
  - Simple design example
  - Boolean theorems
  - Comparison of Boolean functions
  - Canonical and standard forms

# Boolean Functions

- Boolean elements and operators can express a function
  - Value of function determined by value of variables
  - Complete function is evaluated for all possible values
- Function can be represented in a standard form as a Sum of Products (SOP) or by a truth table

- E.g.,  $F = x + y'z$

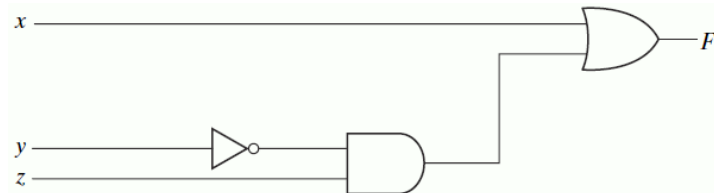
Can you think of an application that is modeled by  $F$ ?

- Evaluate in steps
  - E.g., first  $y'$ , then  $y'z$ , etc.
- A special type of product term is called *minterm*

x	y	z	y'	y'z	x+y'z
0	0	0	1	0	0
0	0	1	1	1	1
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	1	0	1
1	0	1	1	1	1
1	1	0	0	0	1
1	1	1	0	0	1

# Logic Circuit Diagram of Function

- Logic function can be expressed as circuit diagram
  - Direct translation from algebraic expression
- Example:  $F = x + y'z$



- Problem: possibly multiple equivalent algebraic expressions
  - Difficult to compare Boolean functions

# Design Problem

- Design the control logic for a fan in a greenhouse.

The logic must sense three environmental conditions:

- It is raining outside (variable  $x$ )
- It is humid inside (variable  $y$ )
- It is hot inside (variable  $z$ )

The fan should turn on when

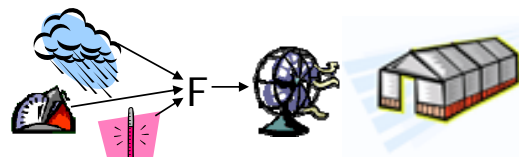
- It is not raining AND (it is humid) AND (it is hot)

$$x' y z$$

OR

- It is not humid AND { (it is raining) OR (it is not raining AND it is hot)}

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



- Create a Boolean function that controls the fan.
  - $F = yx'z + y'(x'z+x)$
  - Other possibilities:  $F = x'y'z+x'yz+xy'$  or  $F = x'z+xy'$
- Question: How to evaluate these solutions (equivalent functions)?

# Boolean Function Representations

Greenhouse example function can be expressed as:

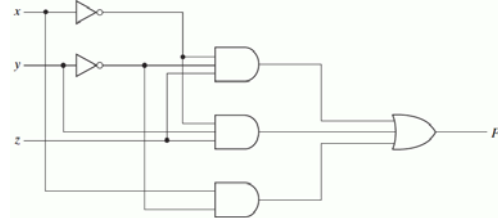
- Analytically, in standard form as sum of products:

$$\begin{aligned}
 yx'z + y'(x'z+x) &= yx'z + y'x'z + y'x \\
 &= x'yz + x'y'z + xy'(z+z') \\
 &= x'yz + x'y'z + xy'z + xy'z'
 \end{aligned}$$

- As a truth table (canonical form)

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

- As a logic network



- How to derive different solutions ?
- How to evaluate these solutions ?

# Boolean Theorems

- Huntington's postulates define some rules

Post. 1: closure
Post. 2: (a) $x+0=x$ , (b) $x \cdot 1=x$
Post. 3: (a) $x+y=y+x$ , (b) $x \cdot y=y \cdot x$
Post. 4: (a) $x(y+z) = xy+xz$ , (b) $x+yz = (x+y)(x+z)$
Post. 5: (a) $x+x'=1$ , (b) $x \cdot x'=0$

- Need more rules to modify algebraic expressions

- Theorems that are derived from postulates

- What is a theorem?

- A formula or statement that is derived from postulates (or other proven theorems)

- Basic theorems of Boolean algebra

- Theorem 1 (a):  $x + x = x$  (b):  $x \cdot x = x$
- Looks straightforward, but needs to be proven !

## Proof of $x+x=x$

- We can only use Huntington postulates:

Huntington postulates:

**Post. 2:** (a)  $x+0=x$ , (b)  $x \cdot 1=x$   
**Post. 3:** (a)  $x+y=y+x$ , (b)  $x \cdot y=y \cdot x$   
**Post. 4:** (a)  $x(y+z) = xy+xz$ ,  
(b)  $x+yz = (x+y)(x+z)$   
**Post. 5:** (a)  $x+x'=1$ , (b)  $x \cdot x'=0$

- Show that  $x+x=x$ .

$$\begin{aligned}x+x &= (x+x) \cdot 1 && \text{by 2(b)} \\ &= (x+x)(x+x') && \text{by 5(a)} \\ &= x+x \cdot x' && \text{by 4(b)} \\ &= x+0 && \text{by 5(b)} \\ &= x && \text{by 2(a)} \\ &\text{Q.E.D.}\end{aligned}$$

- We can now use Theorem 1(a) in future proofs

## Proof of $x \cdot x=x$

- Similar to previous proof

Huntington postulates:

**Post. 2:** (a)  $x+0=x$ , (b)  $x \cdot 1=x$   
**Post. 3:** (a)  $x+y=y+x$ , (b)  $x \cdot y=y \cdot x$   
**Post. 4:** (a)  $x(y+z) = xy+xz$ ,  
(b)  $x+yz = (x+y)(x+z)$   
**Post. 5:** (a)  $x+x'=1$ , (b)  $x \cdot x'=0$   
**Th. 1:** (a)  $x+x=x$

- Show that  $x \cdot x = x$ .

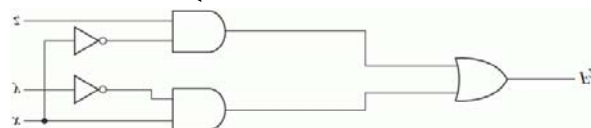
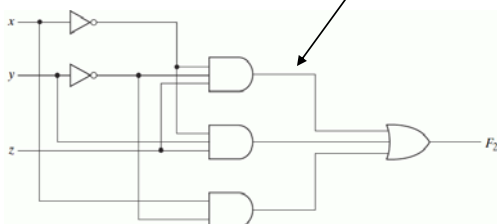
$$\begin{aligned}x \cdot x &= x \cdot x+0 && \text{by 2(a)} \\ &= x \cdot x+x \cdot x' && \text{by 5(b)} \\ &= x \cdot (x+x') && \text{by 4(a)} \\ &= x \cdot 1 && \text{by 5(a)} \\ &= x && \text{by 2(b)} \\ &\text{Q.E.D.}\end{aligned}$$

# Boolean Theorems

- Other theorems that can be derived
  - Theorem 1(a):  $x+x=x$
  - Theorem 1(b):  $x \cdot x=x$
  - Theorem 2(a):  $x+1=1$
  - Theorem 2(b):  $x \cdot 0=0$
  - Theorem 3:  $(x')'=x$
  - Theorem 4(a):  $x+(y+z)=(x+y)+z$
  - Theorem 4(b):  $x(yz)=(xy)z$
  - Theorem 6(a):  $x+xy=x$
  - Theorem 6(b):  $x(x+y)=x$
- Theorem 5: DeMorgan's theorem (duality)
  - Theorem 5(a):  $(x+y)' = x'y'$
  - Theorem 5(b):  $(xy)' = x'+y'$

# Comparison of Boolean Functions

- How to compare expressions?
  - From the Greenhouse Example:
    - Is  $yx'z + y'(x'z+x) = x'z+xy'$  ? (are these functions *equivalent*?)
- $$\begin{aligned}
 yx'z + y'(x'z+x) &= yx'z+y'x'z+y'x \\
 &= (y+y')x'z+y'x \\
 &= x'z+xy'
 \end{aligned}$$
- Yes, both expressions describe the same function (equivalent)
- Problem: hard to compare algebraic expressions
    - Easier with gate level comparison?



# Comparison of Boolean functions

- Comparison by algebraic expression
  - Good: can handle many variables
  - Bad: not clear how to get from one expression to another
- Comparison on gate level
  - Bad: same as algebraic expression
  - Bad: hard to modify for comparison
- Comparison by truth table
  - Good: Only one representation in truth table
  - Bad: cumbersome for 4 or more variables
  - Bad: hard to derive gate level implementation
- Comparison with canonical form
  - Good: standardized form for algebraic expression
  - Bad: not necessarily solution with least number of gates

# Canonical Form

- A form is *canonical* if representation of a function in this form is unique
  - Truth table is canonical representation
  - Uses *minterms* as basic component
- A minterm is a product (*ANDing*) of all variables
  - Each variable of function appears in minterm
  - Variable can appear in normal form ( $x$ ) or in complemented form ( $x'$ )
- Example: minterms for a 3-variable function
- All algebraic expressions can be converted into canonical form

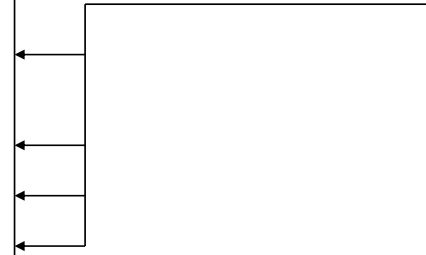
$x$	$y$	$z$	minterm	designation
0	0	0	$x'y'z'$	$m_0$
0	0	1	$x'y'z$	$m_1$
0	1	0	$x'yz'$	$m_2$
0	1	1	$x'yz$	$m_3$
1	0	0	$xy'z'$	$m_4$
1	0	1	$xy'z$	$m_5$
1	1	0	$xyz'$	$m_6$
1	1	1	$xyz$	$m_7$

# Canonical Form

- Function is expressed as sum of minterms

Greenhouse example:  $yx'z + y'(x'z+x) = yx'z + y'x'z + y'x$   
 $= x'y'z + x'y'z + xy'(z+z')$   
 $= x'y'z + x'y'z + xy'z + xy'z'$   
 $= m_3 + m_1 + m_5 + m_4$

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



- Function expression = sum of all minterms where  $F=1$

# Canonical Form

- Canonical form: sum of minterms for  $F=1$

- Example:  $F=A+B'C$

- Expansion of minterms:  
 $A = ABC + ABC' + AB'C + AB'C'$   
 $B'C = AB'C + A'B'C$

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

- Alternate forms:

- $F = m_1 + m_4 + m_5 + m_6 + m_7$
- $F = \sum(1, 4, 5, 6, 7)$

- Is this the only canonical form?

- No, other forms exist (dual form, K-maps, etc.)

A	B	C	F	minterm	designation
0	0	0	0	$A'B'C'$	$m_0$
0	0	1	1	$A'B'C$	$m_1$
0	1	0	0	$A'BC'$	$m_2$
0	1	1	0	$A'BC$	$m_3$
1	0	0	1	$AB'C'$	$m_4$
1	0	1	1	$AB'C$	$m_5$
1	1	0	1	$ABC'$	$m_6$
1	1	1	1	$ABC$	$m_7$

# Summary

- **Boolean algebra**
  - Boolean postulates and theorems
- **Boolean functions**
  - Comparison of Boolean functions
  - Standard form (sum of products)
  - Canonical forms (sum of minterms, truth table)
- **Next ( to finish Chapter 2)**
  - Duality: DeMorgan's theorem
  - Dual canonical form (product of maxterms)
  - Digital logic gates and integrated circuits