# Booliean Algebra and Logic Gates

2019. 3. 11.

K-S. Sohn

## Contents



## Some Definitions

- Set
  - any collection of objects, usually having a common property
- Binary operator
  - in abstract symbol: *, ⊙
  - in real symbol: +, ·
- Postulate
  - the basic assumptions from which it is possible to deduce the rules, theorems, and properties of a mathematical system.
- Field
  - A set of elements, together with operators, each having properties of postulate 1 through 5 and operators combining to give property of postulate 6. ☞ see next slides for the postulates.



## Common Postulates in Algebraic Systems

1. Closure
   - A set S is closed with respect to * if
   - $x*y=c \in S,\ \forall\ x,y \in S$
   - Q.: is + closed in the set of natural numbers? how about -, *, and / ?
2. Associative law
   - An operator * on a set S is associative if
   - $(x*y)*z = x*(y*z)\ \forall x,y,z \in S$
3. Commutative law
   - An operator * is commutative if
   - $x*y=y*x\ \forall x,y \in S$

## Common Postulates in Algebraic Systems

4. Identity element
   - An element *e* is the identity element over the operator * if:
   - $e*x=x*e=x \ \forall x \in S$
   - Ex.: *e* = 0 for a binary operator + on the set of integers *I*
5. Inverse
   - *y* is the inverse of x over the operator *, whenever
   - $x*y=e, \ \forall \ x,y \in S$.
6. Distributive law
   - An operator * is distributive over $\odot$ whenever
   - $x*(y \odot z)=(x*y) \odot (x*z)$



## Field of Real Numbers

- The field of real numbers is the basis for arithmetic and ordinary algebra.
  - The binary operator + defines addition.
  - The additive identity is 0.
  - The additive inverse defines subtraction.
  - The binary operator ?defines multiplication.
  - The multiplicative identity is 1.
  - For a ≠ 0, the multiplicative inverse of a = 1/a
  - defines division (i.e., a ?1/a = 1).
  - The only distributive law applicable is that of · over +:
  - a · (b + c) = (a · b) + (a · c)



## Huntington Postulates

- p.1 -- Closure
  - *B* is closed with + and ·
- p.2 -- Identity element
  - (a). 0 for the operator +
  - (b). 1 for the operator ·
- p.3 -- Commutative
  - (a). $x+y=y+x$
  - (b). $x \cdot y=y \cdot x$



## Huntington postulate

- p.4 -- Distributive
  - (a). · is distributive over + : $x \cdot (y+z)=x \cdot y+x \cdot z$
  - (b). + is distributive over · : $x+(y \cdot z)=(x+y) \cdot (x+z)$
- p.5 -- Complement
  - (a). $\exists x'$ such that $x+x'=1$,
  - (b). $\exists$ x' such that $x \cdot x'=0$, $x \in B$
- p.6 -- Cardinality
  - $|B| \geq 2$, where $|B|$ represents the number of elements of *B*

## Differences between H.P. and O.A.

- Associative law
  - the Huntington Postulate(H.P.) has no associative law
  - but can be derived from other postulate for Boolean algebra(B.A.)
- Distibutive law
  - In H.P., + is distributive over ·
  - but not for ordinary algebra(O.A.)
- Inverse
  - H.P. has No additive nor multiplicative inverse.
  - B.A. has no subtraction or division operations.



## Differences between H.P. and O.A.

- Complement operator
  - O.A. has no complement operator such as x'
  - Why were complements mentioned in the number system?
- Set elements
  - In O.A. B is the infinite set of the real numbers
  - where, in B.A., B is undefined but finite set
  - but in the two?valued Boolean algebra defined next (and of interest in our subsequent use of that algebra), B is defined as a set with only two elements, 0 and 1.



## Boolean Algebraic System

- Element set : *B*
  - Two-valued boolean algebra : *B* = { 0, 1 }
  - or *B* is undefined
- Binary operators : +, ·
  - +: OR operator
  - · : AND operator
  - take symbols only for conventional familiarity
- Unary operator: '
  - ' : NOT (or complement) operator



## Two-Valued Boolean Algebra

- Rules of B.A. operations
  - +, ·, and '

| $x$ | $y$ | $x \cdot y$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

| $x$ | $y$ | $x + y$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

| $x$ | $x'$ |
|---|---|
| 0 | 1 |
| 1 | 0 |

## H.P. test for 2-valued B.A.

- Closure
  - 0 + 0 = 0
  - 0 + 1 = 1 + 0 = 1
  - identity elements are 0 and 1 respectively for for + and ·.
- Commutativeness

| $x$ | $y$ | $x \cdot y$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

| $x$ | $y$ | $x + y$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

| $x$ | $x'$ |
|---|---|
| 0 | 1 |
| 1 | 0 |

  - why?



## H.P. test for 2-valued B.A.

- Distributiveness
  - · over +

| $x$ | $y$ | $z$ | $y+z$ | $x\cdot(y+z)$ | $x\cdot y$ | $x\cdot z$ | $(x\cdot y)+(x\cdot z)$ |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

  - + over ·



## H.P. test for 2-valued B.A.

- Complements existence
  - x + x' = 1, since 0 + 0' = 0 + 1 = 1 and 1 + 1' = 1 + 0 = 1.
  - x · x' = 0, since 0 · 0' = 0 · 1 = 0 and 1 · 1' = 1 · 0 = 0.



## Basic Theorems and Properties of B.A.

- Duality
  - Some Huntington postulates consist of 2 parts of sub-postulate deginated for + and • operator, respectively.
  - One part may be obtained from the other if the binary , operator and identity element are interchanged.
  - i.e., 1 ⇆ 0, + ⇆ • or OR ⇆ AND
  - Ex.: $x \cdot (x+y) = x \rightarrow x + (x \cdot y) = x$

## Basic Theorems and Properties of B.A.

- Summary of Postulates and Theorems

| | | | | |
|---|---|---|---|---|
| Postulate 2 | (a) | $x + 0 = x$ | (b) | $x \cdot 1 = x$ |
| Postulate 5 | (a) | $x + x' = 1$ | (b) | $x \cdot x' = 0$ |
| Theorem 1 | (a) | $x + x = x$ | (b) | $x \cdot x = x$ |
| Theorem 2 | (a) | $x + 1 = 1$ | (b) | $x \cdot 0 = 0$ |
| Theorem 3, involution | | $(x')' = x$ | | |
| Postulate 3, commutative | (a) | $x + y = y + x$ | (b) | $xy = yx$ |
| Theorem 4, associative | (a) | $x + (y + z) = (x + y) + z$ | (b) | $x(yz) = (xy)z$ |
| Postulate 4, distributive | (a) | $x(y + z) = xy + xz$ | (b) | $x + yz = (x + y)(x + z)$ |
| Theorem 5, DeMorgan | (a) | $(x + y)' = x'y'$ | (b) | $(xy)' = x' + y'$ |
| Theorem 6, absorption | (a) | $x + xy = x$ | (b) | $x(x + y) = x$ |



## Theorem 1

- (a) x + x = x

| Statement | Justification |
|---|---|
| $x + x = (x + x) \cdot 1$ | postulate 2(b) |
| $= (x + x)(x + x')$ | 5(a) |
| $= x + xx'$ | 4(b) |
| $= x + 0$ | 5(b) |
| $= x$ | 2(a) |



## Theorem 1

- (b) $x \cdot x = x.$

| Statement | Justification |
|---|---|
| $x \cdot x = xx + 0$ | postulate 2(a) |
| $= xx + xx'$ | 5(b) |
| $= x(x + x')$ | 4(a) |
| $= x \cdot 1$ | 5(a) |
| $= x$ | 2(b) |



## Theorem 2

- (a) $x + 1 = 1.$

| Statement | Justification |
|---|---|
| $x + 1 = 1 \cdot (x + 1)$ | postulate 2(b) |
| $= (x + x')(x + 1)$ | 5(a) |
| $= x + x' \cdot 1$ | 4(b) |
| $= x + x'$ | 2(b) |
| $= 1$ | 5(a) |

- (b) x · 0 = 0
  - by duality

## Theorem 3: Involution

- (x')' = x
  - From H.P. 5, x + x' = 1 and x · x' = 0 are defined.
  - These say that the complement of x' is x.
  - ⇒ (x')' = x, also.



## Theorem 6: Absorption

- (a) $x + xy = x.$

| Statement | Justification |
|---|---|
| $x + xy = x \cdot 1 + xy$ | postulate 2(b) |
| $= x(1 + y)$ | 4(a) |
| $= x(y + 1)$ | 3(a) |
| $= x \cdot 1$ | 2(a) |
| $= x$ | 2(b) |

- (b) x(x+y) = y
  - by duality



## Theorem 5: DeMorgan

- (a) (x + y)' = x'y'
  - by truth table

| $x$ | $y$ | $x + y$ | $(x + y)'$ |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 |

| $x'$ | $y'$ | $x'y'$ |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 0 |

- (b) (xy)' = x' + y'
  - by duality



## Operator Predecence

1. Parentheses
2. NOT
3. AND
4. OR

- Example
  - xy' + z
  - (xy + z)'

# Boolean Functions

- A Boolean function
  - binary variables
  - binary operators OR and AND
  - unary operator NOT
  - parentheses
- Examples
  - F1= x + y' z
  - F2 = x' y' z + x' y z + x y'



# Example: F = x + y'z

- Truth table
  - Combination of variables ( input )
  - Functions( output ) for each combination

**Table 2-2**
*Truth Tables for $F_1$ and $F_2$*

| x | y | z | $F_1$ | $F_2$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 0 |



# Example: F = x + y'z

- Logic circuit diagram
  - Operators → gates
  - Variables → input to the gates
  - Output of the gate → input to another gates

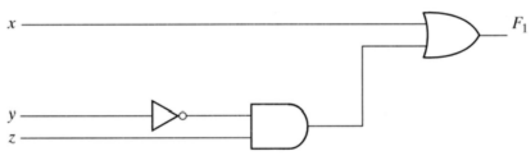



**FIGURE 2-1**
Gate implementation of $F_1 = x + y'z$



# Example 2: F2 = x'y'z + x'y z + xy'

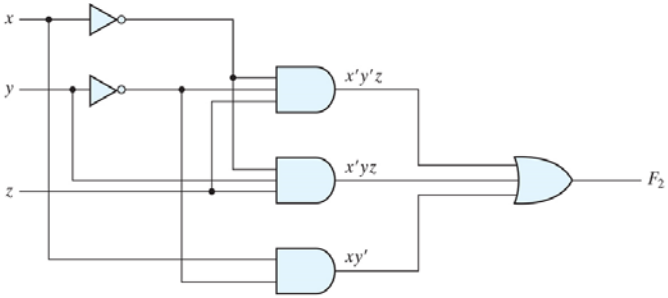
- Truth table

**Table 2.2**
*Truth Tables for $F_1$ and $F_2$*

| x | y | z | $F_1$ | $F_2$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 0 |

# Example 2: F2 = x'y'z + x'y z + xy'

- Logic diagram

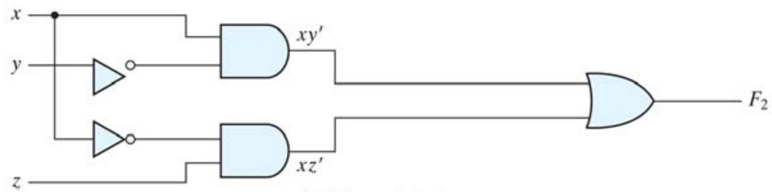



(a) $F_2 = x'y'z + x'yz + xy'$



# Example 2: F2 = x'y'z + x'y z + xy'

- Simplified logic diagram



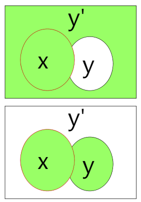

(b) $F_2 = xy' + x'z$



# Simplifying Boolean Functions

- Algebraic manipulation
  - Literal : a single variable within a term
  - Reducing terms and/or literals → simpler circuit
  - Examples : minimum number of literals

$$x(x'+y) = xx' + xy = xy$$
$$x + x'y = (x+x')(x+y) = x+y$$
$$(x+y)(x+y') = xx + xy' + yx + yy' = x(1+y+y') = x$$
$$xy + x'z + yz = xy + x'z + yz(x+x') = xy + x'z + xyz + x'yz$$
duality
$$= xy(1+z) + x'z(1+y) = xy + x'z$$
$$(x+y)(x'+z)(y+z) = (x+y)(x'+z)$$
duality






# Complement of Function

- Generalized DeMorgan's Theorem
  - $(A+B+C+D+\ldots+F)' = A'B'C'D'\ldots F'$
  - $(ABCD\ldots F)' = A'+B'+C'+D'+\ldots+F'$
  - Examples
    - $F_1=x'yz'+x'y'z \leftrightarrow F_1'=(x'yz')'(x'y'z)'=(x+y'+z)(x+y+z')$
    - $F_2=x(y'z'+yz) \leftrightarrow F_2'=x'+(y'z'+yz)'=x'+(y'z')'(yz)'=x'+(y+z)(y'+z')$
- Complement procedure
  - step-1. Take the dual of the function
  - step-2. Complement each literal
  - Example
    - $F_1=x'yz'+x'y'z \leftrightarrow F_1'=(x+y'+z)(x+y+z')$
    - $F_2=x(y'z'+yz) \leftrightarrow F_2'=x'+(y+z)(y'+z')$

## Minterm and Maxterm

- Minterm (standard product)
  - AND term
  - $n$ variables → $2^n$ minterms
- Maxterm (standard sum)
  - OR term
  - $n$ variables → $2^n$ maxterms
- Example($n$=3)

| | | | 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 | $xyz$ | $m_7$ | $x'+y'+z'$ | $M_7$ |



## Canonical Form

- Sum of minterm
  - Boolean function can be expressed as a sum of minterms each of which makes the function produce 1.
- Product of maxterm
  - Complement of the boolean function can be expressed as a sum of minterms each of which makes the function produce 0.
- Drawbacks
  - Not minimized literals in the canonical forms
  - ⇒ Standard form is required



## Truth Table to Boolean Algebra

- Truth table

| $x$ | $y$ | $z$ | Function $f_1$ | Function $f_2$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

- Sum of minterm
  - $f_1=x'y'z+xy'z'+xyz = m_1+m_4+m_7 = \sum(1,4,7)$
  - $f_2=x'yz+xy'z+xyz'+xyz = m_3+m_5+m_6+m_7 = \sum(3,5,6,7)$



## Truth Table to Boolean Algebra

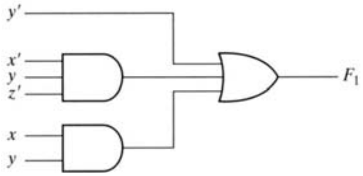
- Product of maxterm
  - $f_1'=x'y'z'+x'yz'+x'yz+xy'z+xyz'$

$$\rightarrow f_1=(x+y+z)(x+y'+z)(x+y'+z')(x'+y+z')(x'+y'+z)$$

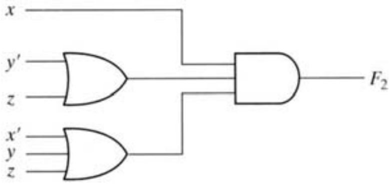
$$= M_0 \cdot M_2 \cdot M_3 \cdot M_5 \cdot M_6 = \Pi(0,2,3,5,6)$$

# Standard Forms

- Sum of products
  - Each term has one or more literals
  - OR-ing the product terms
  - 2-level circuit implementation
  - e.g. $F_1=y'+xy+x'yz'$



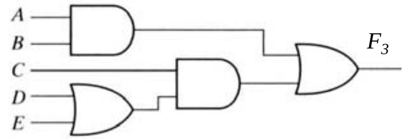

- Product of sums
  - Each term has one or more literals
  - AND-ing the sum terms
  - 2-level circuit implementation
  - e.g. $F_2=x(y'+z)(x'+y+z')$



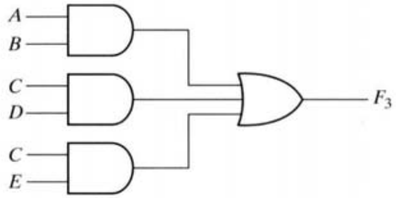



# Non-standard Form

- Form
  - Mixture of product of sums and sum of products
  - Multi-level circuit implementation
    - ⇒ Larger amount of delay




- Non-standard form ⇒ Standard form
  - Distributive law
  - Example
    - F3=AB+C(D+E)
    - F3=AB+C(D+E)=AB+CD+CE





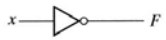
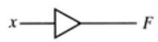

# Other Logic Operations

***Boolean Expressions for the 16 Functions of Two Variables***

| Boolean functions | Operator symbol | Name | Comments |
|---|---|---|---|
| $F_0 = 0$ | | Null | Binary constant 0 |
| $F_1 = xy$ | $x \cdot y$ | AND | $x$ and $y$ |
| $F_2 = xy'$ | $x/y$ | Inhibition | $x$, but not $y$ |
| $F_3 = x$ | | Transfer | $x$ |
| $F_4 = x'y$ | $y/x$ | Inhibition | $y$, but not $x$ |
| $F_5 = y$ | | Transfer | $y$ |
| $F_6 = xy' + x'y$ | $x \oplus y$ | Exclusive-OR | $x$ or $y$, but not both |
| $F_7 = x + y$ | $x + y$ | OR | $x$ or $y$ |
| $F_8 = (x + y)'$ | $x \downarrow y$ | NOR | Not-OR |
| $F_9 = xy + x'y'$ | $(x \oplus y)'$ | Equivalence | $x$ equals $y$ |
| $F_{10} = y'$ | $y'$ | Complement | Not $y$ |
| $F_{11} = x + y'$ | $x \subset y$ | Implication | If $y$, then $x$ |
| $F_{12} = x'$ | $x'$ | Complement | Not $x$ |
| $F_{13} = x' + y$ | $x \supset y$ | Implication | If $x$, then $y$ |
| $F_{14} = (xy)'$ | $x \uparrow y$ | NAND | Not-AND |
| $F_{15} = 1$ | | Identity | Binary constant 1 |

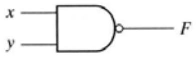


# Digital Logic Gates

| Name | Graphic symbol | Algebraic function | Truth table |
|---|---|---|---|
| AND |  | $F = xy$ | x y \| F<br>0 0 \| 0<br>0 1 \| 0<br>1 0 \| 0<br>1 1 \| 1 |
| OR |  | $F = x + y$ | x y \| F<br>0 0 \| 0<br>0 1 \| 1<br>1 0 \| 1<br>1 1 \| 1 |
| Inverter |  | $F = x'$ | x \| F<br>0 \| 1<br>1 \| 0 |
| Buffer |  | $F = x$ | x \| F<br>0 \| 0<br>1 \| 1 |

## Digital Logic Gates

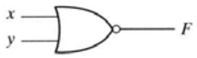
**NAND**




$F = (xy)'$

| x | y | F |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

**NOR**




$F = (x + y)'$

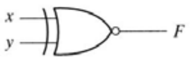
| x | y | F |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

**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 |

**Exclusive-NOR or equivalence**




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

| x | y | F |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |



# Discussion~~~