CS2100 COMPUTER ORGANISATION

http://www.comp.nus.edu.sg/~cs2100/

Lecture #13

Boolean Algebra

NUS National University of Singapore | School of Computing

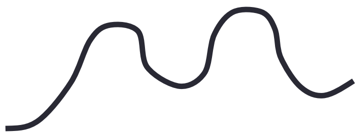
Lecture #13: Boolean Algebra

1. Digital Circuits
2. Boolean Algebra
3. Truth Table
4. Precedence of Operators
5. Laws of Boolean Algebra
6. Duality
7. Theorems
8. Boolean Functions
9. Complement Functions
10. Standard Forms
11. Minterms and Maxterms
12. Canonical Forms:
    Sum-of-Minterms and Product-of-Maxterms

1. Digital Circuits (1/2)

- Two voltage levels
  - High/true/1/asserted
  - Low/false/0/deasserted

Signals in digital circuit



Signals in analog circuit

- Advantages of digital circuits over analog circuits
  - More reliable (simpler circuits, less noise-prone)
  - Specified accuracy (determinable)
  - Abstraction can be applied using simple mathematical model – Boolean Algebra
  - Ease design, analysis and simplification of digital circuit – Digital Logic Design

1. Digital Circuits (2/2)

- Combinational: no memory, output depends solely on the input
  - Gates
  - Decoders, multiplexers
  - Adders, multipliers
- Sequential: with memory, output depends on both input and current state
  - Counters, registers
  - Memories

2. Boolean Algebra

- Boolean values:
  - True (T or 1)
  - False (F or 0)

- Connectives
  - Conjunction (AND)
    - A · B; A ∧ B
  - Disjunction (OR)
    - A + B; A ∨ B
  - Negation (NOT)
    - A'; $\overline{A}$; ¬A;

In CS2100, we use the symbols 1 for true, 0 for false, · for AND, + for OR, and ' for negation (you may use the accent bar). Please follow.

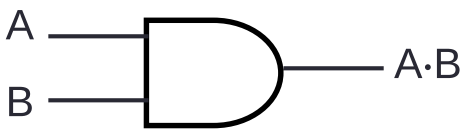
- Truth tables

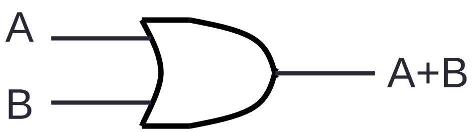
A	B	A · B
0	0	0
0	1	0
1	0	0
1	1	1

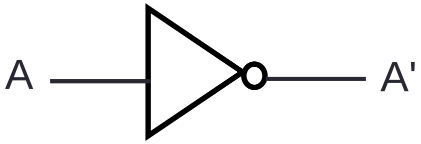
A	B	A + B
0	0	0
0	1	1
1	0	1
1	1	1

A	A'
0	1
1	0

- Logic gates







2. Boolean Algebra: AND

- Do write the AND operator · (instead of omitting it)
  - Example: Write ***a·b*** instead of ***ab***
  - Why? Writing ***ab*** could mean that it is a 2-bit value.

3. Truth Table

- Provide a listing of every possible combination of inputs and its corresponding outputs.
  - Inputs are usually listed in binary sequence.
- Example
  - Truth table with 3 inputs x, y, z and 2 outputs (y + z) and (x · (y + z)).

x	y	z	y + z	x · (y + z)
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	1	0
1	0	0	0	0
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

3. Proof using Truth Table

- Prove: **x · (y + z) = (x · y) + (x · z)**
  - Construct truth table for LHS and RHS

x	y	z	y + z	**x · (y + z)**	x · y	x · z	**(x · y) + (x · 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

  - Check that column for LHS = column for RHS

- DLD page 59 Quick Review Questions Question 3-1.

4. Precedence of Operators

- Precedence from highest to lowest
  - Not (')
  - And (·)
  - Or (+)

Note the difference with CS1231/CS1231S. Here in CS2100, AND has higher precedence than OR.

- Examples:
  - A · B + C = (A · B) + C
  - X + Y' = X + (Y')
  - P + Q' · R = P + ((Q') · R)

Hence, A · B + C is <u>not</u> ambiguous in CS2100.

- Use parenthesis to overwrite precedence. Examples:
  - A · (B + C) [Without parenthesis, it means A·B+C or (A·B)+C]
  - (P + Q)' · R [Without parenthesis, it means P+Q'·R or P+(Q'·R)]

5. Laws of Boolean Algebra

Identity laws	
$A + 0 = 0 + A = A$	$A \cdot 1 = 1 \cdot A = A$
Inverse/complement laws	
$A + A' = 1$	$A \cdot A' = 0$
Commutative laws	
$A + B = B + A$	$A \cdot B = B \cdot A$
Associative laws *	
$A + (B + C) = (A + B) + C$	$A \cdot (B \cdot C) = (A \cdot B) \cdot C$
Distributive laws	
$A \cdot (B + C) = (A \cdot B) + (A \cdot C)$	$A + (B \cdot C) = (A + B) \cdot (A + C)$

* Due to the associative laws, $A + B + C$ is unambiguous. It may be evaluated as $A + (B + C)$ or $(A + B) + C$. Likewise for $A \cdot B \cdot C$.

6. Duality

- If the AND/OR operators and identity elements 0/1 in a Boolean equation are interchanged, it remains valid.
- Example:
  - The dual equation of $a+(b \cdot c)=(a+b) \cdot (a+c)$ is $a \cdot (b+c)=(a \cdot b)+(a \cdot c)$.
- Duality gives free theorems – "two for the price of one", as a Boolean equation is logically equivalent to its dual. So, you prove one theorem and the other comes for free!
- Examples:
  - If $(x+y+z)' = x' \cdot y' \cdot z'$ is valid, then its dual $(x \cdot y \cdot z)' = x'+y'+z'$ is also valid.
  - If $x+1 = 1$ is valid, then its dual $x \cdot 0 = 0$ is also valid.

Do not confuse duality with negation!

7. Theorems

Idempotency	
$X + X = X$	$X \cdot X = X$
One element / Zero element	
$X + 1 = 1$	$X \cdot 0 = 0$
Involution	
$(X')' = X$	
Absorption 1	
$X + X \cdot Y = X$	$X \cdot (X + Y) = X$
Absorption 2	
$X + X' \cdot Y = X + Y$	$X \cdot (X' + Y) = X \cdot Y$
DeMorgans' (can be generalised to more than 2 variables)	
$(X + Y)' = X' \cdot Y'$	$(X \cdot Y)' = X' + Y'$
Consensus	
$X \cdot Y + X' \cdot Z + Y \cdot Z = X \cdot Y + X' \cdot Z$	$(X+Y) \cdot (X'+Z) \cdot (Y+Z) = (X+Y) \cdot (X'+Z)$

7. Proving a Theorem

- Theorems can be proved using truth table, or by algebraic manipulation using other theorems/laws.
- Example: Prove absorption theorem $X + X \cdot Y = X$

$$
\begin{aligned}
X + X \cdot Y &= X \cdot 1 + X \cdot Y \text{ (by identity law)} \\
&= X \cdot (1+Y) \text{ (by distributivity)} \\
&= X \cdot (Y+1) \text{ (by commutativity)} \\
&= X \cdot 1 \text{ (by one element law)} \\
&= X \text{ (by identity law)}
\end{aligned}
$$

- By the principle of duality, we may also cite (<u>without proof</u>) that $X \cdot (X+Y) = X$.

8. Boolean Functions

- Examples of Boolean functions (logic equations):

  F1(x,y,z) = x·y·z'

  F2(x,y,z) = x + y'·z

  F3(x,y,z) = x'·y'·z + x'·y·z + x·y'

  F4(x,y,z) = x·y' + x'·z

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	1	1	1
0	1	0	0	0	0	0
0	1	1	0	0	1	1
1	0	0	0	1	1	1
1	0	1	0	1	1	1
1	1	0	1	1	0	0
1	1	1	0	1	0	0

Therefore, from the Truth Table, F3 = F4

-Proving F3 = F4 using Boolean Algebra

9. Complement Functions

- Given a Boolean function F, the complement of F, denoted as F', is obtained by interchanging 1 with 0 in the function's output values.
- Example: F1 = x·y·z'
- What is F1' ?

(x.y.z')' = x'+y'+(z')' DeMorgan's

= x'+y'+z Involution

x	y	z	F1	F1'
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	0	1
1	0	0	0	1
1	0	1	0	1
1	1	0	1	0
1	1	1	0	1

10. Standard Forms (1/2)

- Certain types of Boolean expressions lead to circuits that are desirable from an implementation viewpoint.
- Two standard forms:
  - Sum-of-Products (SOP)
  - Product-of-Sums (POS)
- Literals
  - A Boolean variable on its own or in its complemented form
  - Examples: (1) x, (2) x', (3) y, (4) y'
- Product term
  - A single literal or a logical product (AND) of several literals
  - Examples: (1) x, (2) x·y·z', (3) A'·B, (4) A·B, (5) d·g'·v·w

10. Standard Forms (2/2)

- Sum term
  - A single literal or a logical sum (OR) of several literals
  - Examples: (1) x, (2) $x+y+z'$, (3) $A'+B$, (4) $A+B$, (5) $c+d+h'+j$
- Sum-of-Products (SOP) expression
  - A product term or a logical sum (OR) of several product terms
  - Examples: (1) x, (2) $x + y \cdot z'$, (3) $x \cdot y' + x' \cdot y \cdot z$, (4) $A \cdot B + A' \cdot B'$,
    (5) $A + B' \cdot C + A \cdot C' + C \cdot D$
- Product-of-Sums (POS) expression
  - A sum term or a logical product (AND) of several sum terms
  - Examples: (1) x, (2) $x \cdot (y+z')$, (3) $(x+y') \cdot (x'+y+z)$,
    (4) $(A+B) \cdot (A'+B')$, (5) $(A+B+C) \cdot D' \cdot (B'+D+E')$
- Every Boolean expression can be expressed in SOP or POS form.
  - DLD page 59 Quick Review Questions Questions 3-2 to 3-5.

Quiz Time!

SOP expr: A product term or a logical sum (OR) of several product terms.
POS expr: A sum term or a logical product (AND) of several sum terms.

- Put the right ticks in the following table.

	Expression	*SOP?*	*POS?*
(1)	X'·Y + X·Y' + X·Y·Z	✓	
(2)	(X+Y')·(X'+Y)·(X'+Z')		✓
(3)	X' + Y + Z	✓	✓
(4)	X·(W' + Y·Z)		
(5)	X·Y·Z'	✓	✓
(6)	W·X'·Y + V·(X·Z + W')		

11. Minterms and Maxterms (1/2)

- A **minterm** of n variables is a <u>product term</u> that contains n literals from all the variables.
  - Example: On 2 variables x and y, the minterms are: x'·y', x'·y, x·y' and x·y
- A **maxterm** of n variables is a <u>sum term</u> that contains n literals from all the variables.
  - Example: On 2 variables x and y, the maxterms are: x'+y', x'+y, x+y' and x+y
- In general, with n variables we have up to 2^n minterms and 2^n maxterms.

11. Minterms and Maxterms (2/2)

- The minterms and maxterms on 2 variables are denoted by m0 to m3 and M0 to M3 respectively.

x	y	Minterms		Maxterms	
		Term	Notation	Term	Notation
0	0	x'·y'	m0	x+y	M0
0	1	x'·y	m1	x+y'	M1
1	0	x·y'	m2	x'+y	M2
1	1	x·y	m3	x'+y'	M3

- Important fact: Each minterm is the complement of its corresponding maxterm. Likwise, each maxterm is the complement of its corresponding minterm.
  - Example: m2 = x·y'

    m2' = (x·y')' = x' + (y')' = x' + y = M2

M2

Quiz Time Again!

- Ability to convert minterms and maxterms from its Boolean expression to its notation (and vice versa) is important.
- Test yourself with the following quiz, assuming that you are given a Boolean function on 4 variables A, B, C, D.

Minterm

	Boolean expression	***Minterm notation***
(1)	A'·B'·C·D	m3
(2)	A.B'.C.D'	m10
(3)	A.B'.C.D	m11
(4)	A·B·C·D'	m14
(5)	A·B'·C'·D	m9

Maxterm

	Boolean expression	***Maxterm notation***
(1)	A+B+C'+D'	M3
(2)	A'+B'+C+D'	M13
(3)	A+B+C+D	M0
(4)	A+B+C'+D	M2
(5)	A'+B+C+D'	M9

12. Canonical Forms

- Canonical/normal form: a unique form of representation.
  - Sum-of-minterms = Canonical sum-of-products
  - Product-of-maxterms = Canonical product-of-sums

12.1 Sum-of-Minterms

- Given a truth table, example:

x	y	z	F1	F2	F3
0	0	0	0	0	0
0	0	1	0	1	1
0	1	0	0	0	0
0	1	1	0	0	1
1	0	0	0	1	1
1	0	1	0	1	1
1	1	0	1	1	0
1	1	1	0	1	0

- Obtain sum-of-minterms expression by gathering the minterms of the function (where output is 1).

F1 = x·y·z' = m6

F2 = x'.y'.z + x.y'.z' +x.y'.z + x.y.z' + x.y.z
= m1 + m4 + m5 + m6 + m7 = Σm(1+4+5+6+7) or Σ m(1,4-7)

F3 = x'.y'.z + x'.y.z + x.y'.z' + x.y'.z
= m1 + m3 + m4 + m5 = Σ m(1,3,4,5) or Σm(1,3-5)

12.2 Product-of-Maxterms

- Given a truth table, example:
- Obtain product-of-maxterms expression by gathering the maxterms of the function (where output is 0).

x	y	z	F1	F2	F3
0	0	0	0	0	0
0	0	1	0	1	1
0	1	0	0	0	0
0	1	1	0	0	1
1	0	0	0	1	1
1	0	1	0	1	1
1	1	0	1	1	0
1	1	1	0	1	0

$F2 = (x+y+z) \cdot (x+y'+z) \cdot (x+y'+z')$
$\quad = M0 \cdot M2 \cdot M3 = \Pi M(0,2,3)$

$F3 = (x+y+z) . (x+y'+z) . (x'+y'+z) . (x'+y'+z')$
$\quad = M0 . M2 . M6 . M7 = \Pi M(0,2,6,7)$

12.3 Conversion of Standard Forms

- We can convert between sum-of-minterms and product-of-maxterms easily
- Example: F2 = Σm(1,4,5,6,7) = ΠM(0,2,3)
- Why? See F2' in truth table.

x	y	z	F2	F2'
0	0	0	0	1
0	0	1	1	0
0	1	0	0	1
0	1	1	0	1
1	0	0	1	0
1	0	1	1	0
1	1	0	1	0
1	1	1	1	0

- F2' = m0 + m2 + m3
  Therefore,
  F2 = (m0 + m2 + m3)'
  = m0' · m2' · m3' (by DeMorgan's)
  = M0 · M2 · M3 (as m*x*' = M*x*)

- Read up DLD section 3.4, pg 57 – 58.
- Quick Review Questions: pg 60 – 61, Q3-6 to 3-13.

End of File