

Mathematics Concepts For Computing

AQ010-3-1-MCFC



A • P • U
ASIA PACIFIC UNIVERSITY
OF TECHNOLOGY & INNOVATION

Chapter - 5

Logic and Boolean Algebra

Topic & Structure of the lesson

- Introduction
- Logical operations
- Representation of If-Then as Or
- Precedence of Connectives
- Boolean Operators
- Laws of the Boolean Algebra
- Karnaugh Maps

Learning Outcomes

On completion of this chapter you should be able to:

- understand mathematical reasoning in order to read, comprehend, and construct mathematical arguments.
- illustrate an understanding of Boolean Algebra.

Introduction-Logic

- **In mathematics**, use logic
 - to state observation
 - to define concept
 - to formalize theories
 - To derive conclusions from existing information
 - To convince others of these conclusions.
- **In computer science**, use logic
 - to develop languages, to design digital circuits
 - form basis for learning how to reason clearly
 - to model the situations we encounter as professionals, in order to reason about them formally



- A **proposition** is a **declarative sentence** that is either **true** (denoted either T or 1) or **false** (denoted either F or 0). Cannot be partially true or partially false. Cannot be both true and false.
- Notation: **Variables** are used to **represent propositions**. The most common variables used are ***p***, ***q***, and ***r***.
- Example:
 - p : Elephants can fly. (F)
 - q : $1 + 1 = 2$ (T)

Examples of propositions

- a) "Listen!" is **not a proposition**.
- b) "What time is it?" is **not a proposition**.
- c) " $x + 2 = 2$ and $x + y = z$ " are **not a proposition**.
- d) " $x + 2 = x^2$ when $x = 2$ " is a **proposition**.
- Propositional variables can be combined by logical connectives to obtain **compound proposition**.



Proposition

- A proposition is a declarative sentence (that is, a sentence that declares a fact) that is either true or false, but not both.
- Example of declarative sentences which are propositions.
 1. Kathmandu is the capital of the Nepal. ***True proposition***
 2. Sun rises in west. ***False proposition***
 3. $5 + 1 = 6$. ***True proposition***
 4. $7 + 2 = 8$ ***False proposition***
 5. Read this carefully. ***Not proposition.***



Proposition

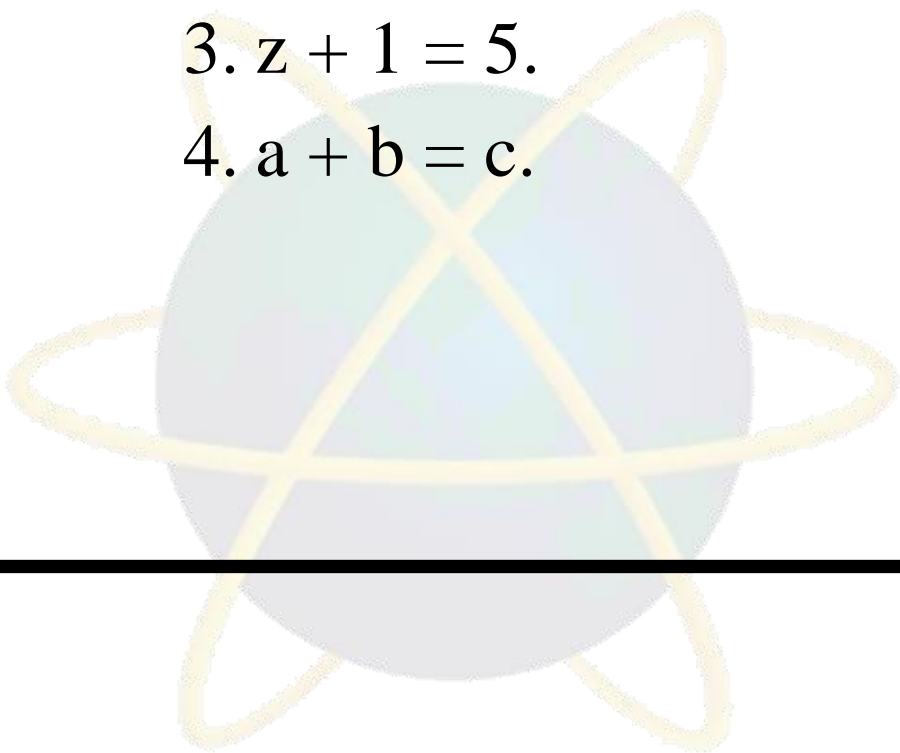
- Example 2: Consider the following sentences.

1. What time is it?

2. Study Well.

3. $z + 1 = 5$.

4. $a + b = c$.



Logical Operation

- **Logical operations**

- Logical operations include **comparisons of two data**.
The result of the comparison will be either true or false.
- It **can be represented using truth table**. (T or 1 represents true, and F or 0 represents false)
- The number of rows to the table is determined by all the possible values taken by the propositions involved in the statement.
- If a compound statement contains **n proposition variables**, there will need to be **2^n** rows in the truth table.

Logical Operation

1. Negation
2. Conjunction
2. Disjunction
3. Exclusive Disjunction (XOR)
5. Implication/Conditional
6. Biconditional

Logical Operation

Connectives	Symbol	Name
not	\sim	negation
and	\wedge	conjunction
or	\vee	disjunction
exclusive or	\oplus	exclusive disjunction(xor)
if...then....	\rightarrow	Implication/conditional
if and only if	\leftrightarrow	biconditional



*Propositions using logical operators: **Negation***

- Let p be a proposition. *The negation of p , denoted by “ $\neg p$ ”, is the statement “It is not the case that p .”*
- The proposition $\neg p$ is read “not p .” The truth values of p , and $\neg p$, are opposite to each other.
- **Example 1:** Find the negation of the proposition “Ram’s laptop runs Windows”
 - Solution: “It is not the case that Ram’s laptop runs Windows.”
 - In other way: “Ram’s laptop does not run Windows.”
- **Example 2:** Find the negation of the proposition “Sony’s Smartphone has at least 4GB of RAM”
 - In other way “Sony’s Smartphone does not have at least 4GB of RAM”
 - More simply as “Sony’s Smartphone has less than 4GB of RAM.”



*Propositions using logical operators: **Conjunction***

- Let p and q be propositions. The conjunction of p and q , denoted by $p \wedge q$, is the proposition “ p and q .” The conjunction $p \wedge q$ is true when both p and q are true and is false otherwise.
- **Example:** Find the conjunction of Kathmandu is the capital city of Nepal and Ancient name of Kathmandu is Patan.
 - Suppose p means “Kathmandu is the capital city of Nepal” and q means “Ancient name of Kathmandu is Patan”.
 - Then $p \wedge q$: Kathmandu is the capital city of Nepal and Ancient name of Kathmandu is Patan.



*Propositions using logical operators: **Disjunction***

- Let p and q be propositions. The disjunction of p and q , denoted by $p \vee q$, is the proposition “ p or q .” The disjunction $p \vee q$ is false when both p and q are false and is true otherwise.
- Example: Find the disjunction of the propositions p and q where p is the proposition “Ram’s PC has more than 16 GB free hard disk space” and q is the proposition “The processor in Ram’s PC runs faster than 1 GHz.”
 - Then $p \vee q$: “Ram’s PC has more than 16 GB free hard disk space, or the processor in Ram’s PC runs faster than 1 GHz.”

Logical operators and truth table

- Table 1. The truth table for the **Negation (not)** of a Proposition

p	$\neg p$
T	F
F	T

eg. p : “ Today is Friday.”

$\neg p$: “ Today is not Friday.”

- Def :** A **truth table** displays the relationships between the truth values of propositions.
- Table 2. The truth table for the **Conjunction (and)** of two propositions.

p	q	$p \wedge q$
T	T	T
T	F	F
F	T	F
F	F	F

eg. p : “ Today is Friday.”

q : “ It’s raining today. ”

$p \wedge q$: “ Today is Friday
and it’s raining
today. “

- Table 3. The truth table for the **Disjunction (or)** of two propositions.

p	q	$p \vee q$
T	T	T
T	F	T
F	T	T
F	F	F

eg. p : "Today is Friday. "

q : "It's raining today. "

$p \vee q$: "Today is Friday or
it's raining today. "

- Table 4. The truth table for the **Exclusive or (xor)** of two propositions.

p	q	$p \oplus q$
T	T	F
T	F	T
F	T	T
F	F	F

eg. p : "Today is Friday. "

q : " It's raining today. "

$p \oplus q$: "Either today is Friday
or it's raining today,
but not both."

- Table 5. The truth table for the Implication (**p implies q**)
p (hypothesis) \rightarrow q (conclusion).

p	q	$p \rightarrow q$
T	T	T
T	F	F
F	T	T
F	F	T

eg. p : “ You make more than \$25000 ”

q : “ You must file a tax return. ”

$p \rightarrow q$: “ If you make more ...
then you must ”

- Some of the more common ways of expressing this implication are :

(1) if p, then q

(2) p implies q

(3) p only if q

(4) q, if p

(5) p is sufficient for q

(6) q is necessary for p

(7) q whenever p

Def : In the implication $p \rightarrow q$, p is called the **hypothesis** and q is called the **conclusion**.



*Propositions using logical operators: **Conditional Statement***

- All of the following ways to express this conditional statement: $p \rightarrow q$

1) “if p, then q”

2) “p implies q”

3) “p **only if** q”

4) “q **whenever** p”

5) “q when p”

6) “q **is necessary for** p”

7) “p **is sufficient for** q”

8) “q **is consequent of** p”

p **only if** q

(if $\sim q$, then $\sim p$ which equivalent to if p then q)

p **is sufficient for** q

(in order to get q, it is sufficient to get p) equivalent to
(to get q, (if) get p)

q **is necessary for** p

(in order to get p, it is necessary to get q) equivalent to
(get p only if get q) (if not get q, then not get p)

q **whenever** p

(whenever get p, get q) equivalent to (if get p then get q)

Propositions using logical operators: Conditional Statement

- **Example 1:** “If I am elected, then I will lower taxes.”
 - Let $p =$ I am elected, and $q =$ I will lower taxes .
 - We can write $p \rightarrow q$
- **Example 2:** Hiking is not safe on the trail whenever grizzly bears have been seen in the area and berries are ripe along the trail.
- **Example 3:** Driving over 65 miles per hour is sufficient for getting a speeding ticket.
- **Example 4:** Whenever you get a speeding ticket, you are driving over 65 miles per hour.
- **Example 5:** We will go swimming only if it is sunny

Propositions using logical operators: Conditional Statement

- **Example 2:** *Hiking* is not safe on the trail whenever *grizzly* bears have been seen in the area and *berries* are ripe along the trail. $(G \wedge B) \rightarrow \neg H$
- **Example 3:** *Driving* over 65 miles per hour is sufficient for getting a speeding ticket. $D \rightarrow T$
- **Example 4:** Whenever you get a speeding ticket, you are driving over 65 miles per hour. $T \rightarrow D$
- **Example 5:** We will *go* swimming only if it is *sunny*. $G \rightarrow S$



Teacher tells you:

If you participate in class, then you will get extra points.

- If you participate in class (True) and you get extra points (True), then the teachers' statement is **TRUE**.
- If you participate in class (True) and you do not get extra points (False), then the teacher did not tell the truth and the statement is **FALSE**.
- If you do not participate in class (False), we cannot judge the truth of the teachers' statement. The teacher did not tell you what would happen if you did NOT participate in class (did not break promise). Then, assign **TRUE** to the statement.

- **Def : Compound propositions** are formed from existing propositions using logical operators. (\wedge 、 \vee 、 \oplus 、 \rightarrow)
- Table 6. The truth table for the **Biconditional** $p \leftrightarrow q$
($p \rightarrow q$ and $q \rightarrow p$)

p	q	$p \rightarrow q$	$q \rightarrow p$	$p \leftrightarrow q$
T	T	T	T	T
T	F	F	T	F
F	T	T	F	F
F	F	T	T	T

“p if and only if q”

“p iff q”

“If p then q , and conversely.”

“p is necessary and sufficient for q”

The flight attendant says:

You can take the flight if and only if you buy a ticket.

- If you buy ticket (True), you can take the flight (True). The statement is **TRUE**.
- If you do not buy ticket (False), you cannot take the flight (False). The statement is **TRUE**.
- If you do not buy ticket (False) but you can take the flight (True), this statement is **FALSE**.
- If you buy ticket (True) but cannot take flight (False), this statement is **FALSE**.

Translating English Sentences into Logical Expression

Example : How can the following English sentence be translated into a logical expression ?

“You can access the Internet from campus only if you are a computer science major or you are not a freshman. ”

Sol :

p : “You can access the Internet from campus.”

q : “You are a computer science major.”

r : “You are a freshman.”

$\therefore p$ only if (q or ($\sim r$))

$\Rightarrow p \rightarrow (q \vee (\sim r))$

- **Example :** You cannot ride the roller coaster if you are under 4 feet tall and you are not older than 16 years old.
- **Sol :** q : “ You can ride the roller coaster. “
 r : “ You are under 4 feet tall. “
 s : “ You are older than 16 years old. “
 $\therefore \sim q$ if r and not s
 $\therefore (r \wedge \sim s) \rightarrow \sim q$

Table 8. Precedence of Logical Operators

Operator	Precedence
\neg	1
\wedge	2
\vee	3
\rightarrow	4
\leftrightarrow	5

eg. (1) $p \wedge q \vee r$ means $(p \wedge q) \vee r$

(2) $p \vee q \rightarrow r$ means $(p \vee q) \rightarrow r$

(3) $p \wedge \sim q$ means $p \wedge (\sim q)$

Exercise

Let p and q be the propositions.

p : it is below freezing.

q : it is snowing.

Write the proposition using p and q and logical connectives.

- (a) it is below freezing and snowing.
- (b) it is below freezing but not snowing.
- (c) It is not below freezing and it is not snowing.
- (d) It is either snowing or below freezing (or both).
- (e) If it is below freezing, it is also snowing.
- (f) Either it is below freezing or its snowing, but it is not snowing if it is below freezing.

Exercise

Let p = “It rained last night”,

q = “The sprinklers came on last night,”

r = “The lawn was wet this morning.”

Translate each of the following into English:

$\sim p$ = “It didn’t rain last night.”

$r \wedge \sim p$ = “The lawn was wet this morning, and it didn’t rain last night.”

$\sim r \vee p \vee q$ = “Either the lawn wasn’t wet this morning, or it rained last night, or the sprinklers came on last night.”

Converse, Contrapositive, and Inverse

- The proposition $q \rightarrow p$ is called the **converse** of $p \rightarrow q$.
- The **contrapositive** of $p \rightarrow q$ is the proposition $\neg q \rightarrow \neg p$. The contrapositive always has the same truth value as $p \rightarrow q$.
- The proposition $\neg p \rightarrow \neg q$ is called the **inverse** of $p \rightarrow q$.
- When two compound propositions always have the same truth value, we call them **equivalent**, so that a conditional statement and its contrapositive are equivalent.

Converse, Contrapositive, and Inverse

- What is the contrapositive, the converse, and the inverse of the conditional statement *“The Nepali team wins whenever it is raining.”*
 - Solution: The “q whenever p” is one of the ways to express the conditional statement $p \rightarrow q$, the original statement can be rewritten as “If it is raining, then the Nepali team wins.”
 - Contrapositive of this conditional statement is “If the Nepali team does not win, then it is not raining.”
 - The converse is “If the Nepali team wins, then it is raining.”
 - The inverse is “If it is not raining, then the Nepali team does not win.”

Propositional logic

- **Atomic proposition:** It is single sentence where each symbol stands for proposition that can be true or false.
 - Example: P means “It is hot.” Q means “It is humid.” R means “It is raining.”
- **Compound proposition:** It is constructed from simple sentences using logical operators.
 - $(P \wedge Q) \rightarrow R$ “If it is hot and humid, then it is raining”
- A compound proposition that is always true, no matter what the truth values of the propositional variables that occur in it, *is called a tautology*.
 - Example: $p \vee \neg p$ is always true regardless of the truth value of the proposition p.
- A compound proposition that is always false *is called a contradiction*.
 - Example: $p \wedge \neg p$ is always false regardless of the truth value of the proposition p.



Propositional logic

- A compound proposition that is neither a tautology nor a contradiction is *called a contingency*.
- **Logical Equivalences:** Compound propositions that have the same truth values in all possible cases are called logically equivalent.
- The compound propositions p and q are called logically equivalent if $p \leftrightarrow q$ is a tautology. The notation $p \equiv q$ denotes that p and q are logically equivalent.
- **Note:** The symbol \equiv is not a logical connective, and $p \equiv q$ is not a compound proposition but rather is the statement that $p \leftrightarrow q$ is a tautology. The symbol \leftrightarrow is sometimes used instead of \equiv to denote logical equivalence.

Compound Propositions

- **Compound propositions:** involving any number of propositional variables and logical connectives.
- Construct the truth table for $(P \rightarrow Q) \wedge (Q \rightarrow R)$.

P	Q	R	$P \rightarrow Q$	$Q \rightarrow R$	$(P \rightarrow Q) \wedge (Q \rightarrow R)$
T	T	T	T	T	T
T	T	F	T	F	F
T	F	T	F	T	F
T	F	F	F	T	F
F	T	T	T	T	T
F	T	F	T	F	F
F	F	T	T	T	T
F	F	F	T	T	T

Compound Propositions

- Construct the truth table of the compound proposition $(p \vee \neg q) \rightarrow (p \wedge q)$.

p	q	$\neg q$	$p \vee \neg q$	$p \wedge q$	$(p \vee \neg q) \rightarrow (p \wedge q)$
T	T	F	T	T	T
T	F	T	T	F	F
F	T	F	F	F	T
F	F	T	T	F	F

Propositional logic

- Show that $\neg(p \vee q)$ and $\neg p \wedge \neg q$ are logically equivalent.

• *Solution:*

p	q	$p \vee q$	$\neg(p \vee q)$	$\neg p$	$\neg q$	$\neg p \wedge \neg q$
T	T	T	F	F	F	F
T	F	T	F	F	T	F
F	T	T	F	T	F	F
F	F	F	T	T	T	T

Propositional logic: Class work

- Show that $p \vee (q \wedge r)$ and $(p \vee q) \wedge (p \vee r)$ are logically equivalent.
- Show that $(P \rightarrow Q) \vee (Q \rightarrow P)$ is a Tautology.



Proving Equivalence via Truth Tables

Prove that $p \vee q \equiv \sim(\sim p \wedge \sim q)$.

p	q	$p \vee q$	$\sim p$	$\sim q$	$\sim p \wedge \sim q$	$\sim(\sim p \wedge \sim q)$
F	F	F	T	T	T	F
F	T	T	T	F	F	T
T	F	T	F	T	F	T
T	T	T	F	F	F	T

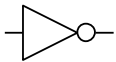
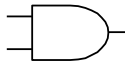


Exercise :

Construct a truth table for each of the following compound proposition :

(a) $\sim p \vee q \wedge r$

(b) $(\sim p \leftrightarrow q) \wedge (q \vee \sim r)$

Some Alternative Notations

Name:	not	and	or	xor	implies	iff
Propositional logic:	\neg	\wedge	\vee	\oplus	\rightarrow	\leftrightarrow
Boolean algebra:	\bar{p}	pq	$+$	\oplus		
C/C++/Java (wordwise):	$!$	$\& \&$	$ $	$!=$		$==$
C/C++/Java (bitwise):	\sim	$\&$	$ $	\wedge		
Logic gates:						

Boolean Algebra

- The rules of logic are used **to design circuits** and **form the basis for Boolean algebra**. The circuits in all electronic devices like computers have inputs that can be identified as either 1 or 0 to produce outputs labeled as 1s and 0s OR switches that are in 'on' or in 'off' position.
- Boolean algebra is a deductive mathematical system closed over the values zero and one (false and true)
- A binary operator accepts a pair of Boolean inputs and produces a single Boolean value.

Boolean Algebra

Boolean algebra deals with

- set of **2 elements** $\{0, 1\}$
- **binary operators:** OR (\vee / +), AND (\wedge / .)
- **unary operator** NOT $\{\neg$ / ' $\}$

Rules:

OR

		\vee / +
0	0	0
0	1	1
1	0	1
1	1	1

AND

		\wedge / .
0	0	0
0	1	0
1	0	0
1	1	1

NOT

	\neg / ' $\}$
0	1
1	0

Boolean Algebra

Rules of precedence are:

- 1) Parentheses/ brackets,
- 2) complement(NOT),
- 3) product (AND),
- 4) sum(OR)

Example:

Show that $(\bar{1} \cdot \bar{0}) + (1 \cdot \bar{0}) = 1$.

Boolean Algebra

- A Boolean function has:
 - At least one Boolean variable,
 - At least one Boolean operator, and
 - At least one input from the set $\{0,1\}$.
- It produces an output that is also a member of the set $\{0,1\}$.

Example

$F(x, y, z) = xy' + z$ is a Boolean function where $xy' + z$ is a Boolean expression. The truth table for $F(x, y, z) = xy' + z$ is

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

The **NOT** operator has highest priority, followed by **AND** and then **OR**.

Exercise

Use a table to express the values of each of these Boolean function.

a) $F(x, y, z) = x'y$

b) $F(x, y, z) = x + yz$

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

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

a)

x	y	z	$\bar{x}y$
1	1	1	0
1	1	0	0
1	0	1	0
1	0	0	0
0	1	1	1
0	1	0	1
0	0	1	0
0	0	0	0

b)

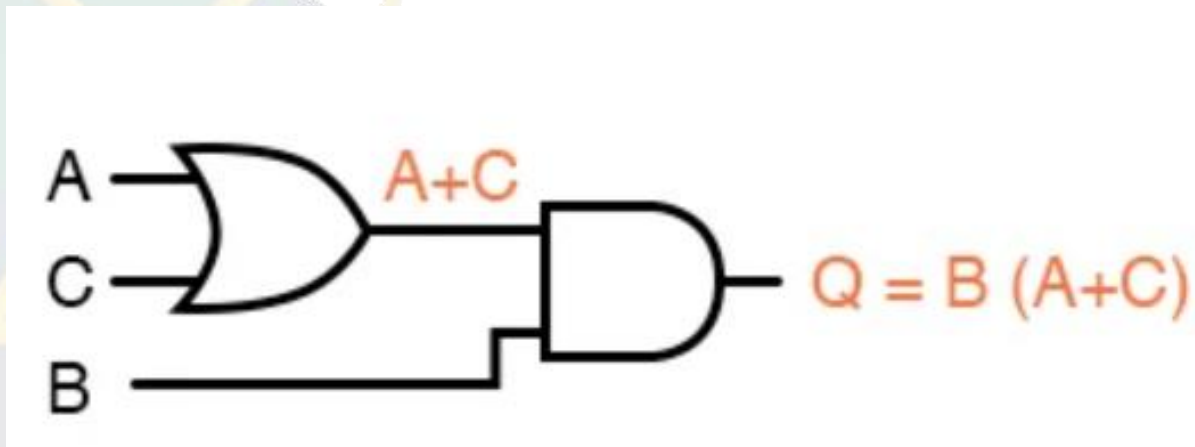
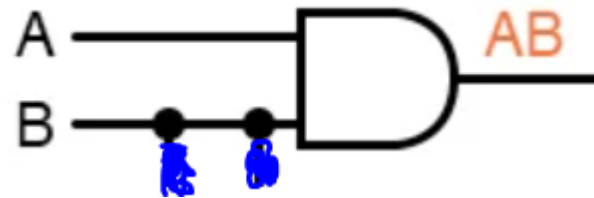
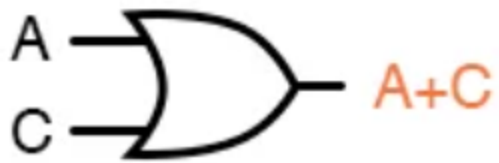
x	y	z	$x + yz$
1	1	1	1
1	1	0	1
1	0	1	1
1	0	0	1
0	1	1	1
0	1	0	0
0	0	1	0
0	0	0	0

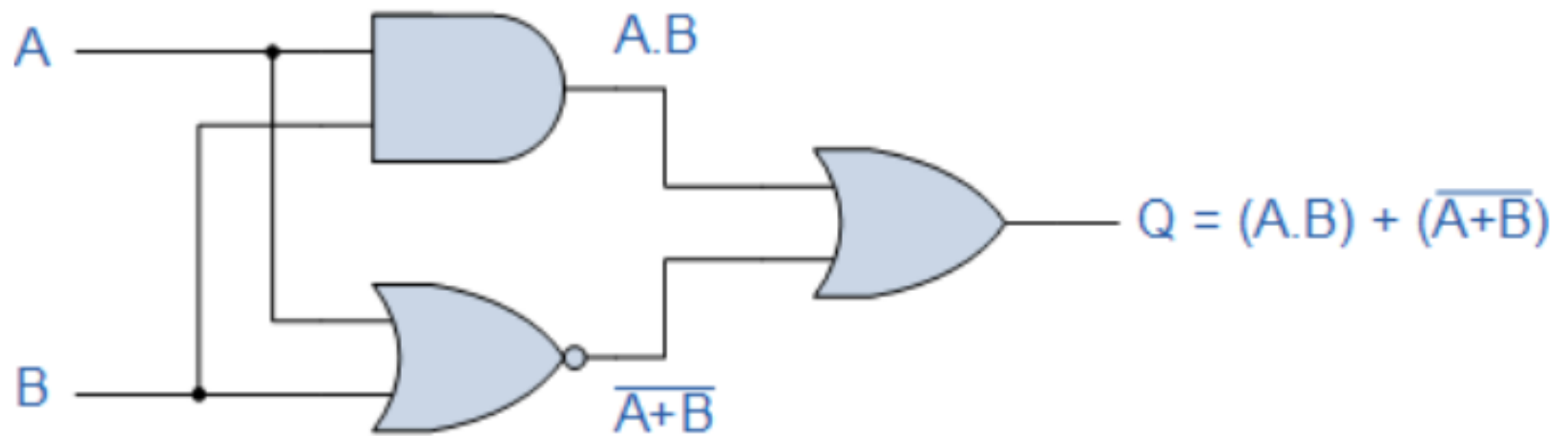
c)

x	y	z	$x\bar{y} + \bar{x}yz$
1	1	1	0
1	1	0	1
1	0	1	1
1	0	0	1
0	1	1	1
0	1	0	1
0	0	1	1
0	0	0	1

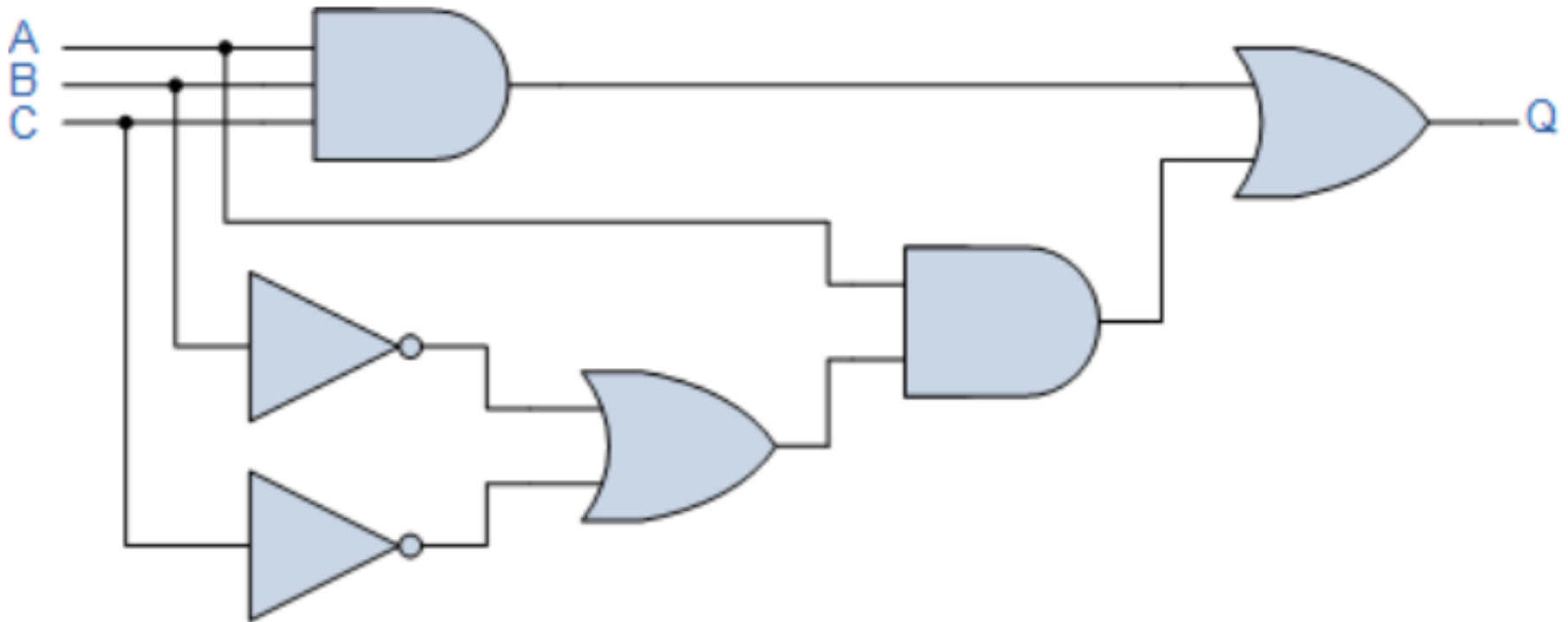
d)

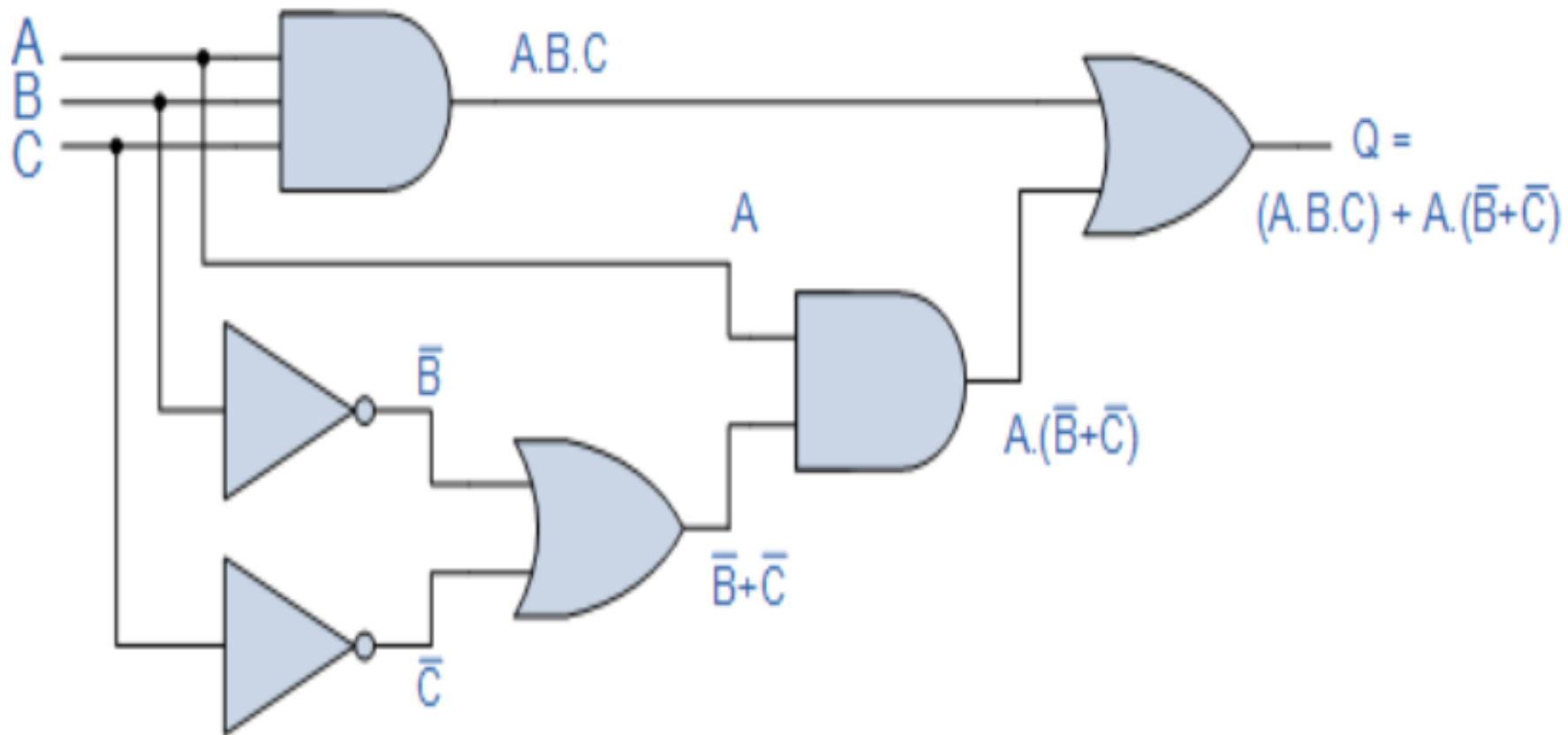
x	y	z	$x(yz + \bar{y}\bar{z})$
1	1	1	1
1	1	0	0
1	0	1	0
1	0	0	1
0	1	1	0
0	1	0	0
0	0	1	0
0	0	0	0





Inputs		Intermediates		Output
B	A	$A.B$	$\overline{A + B}$	Q
0	0	0	1	1
0	1	0	0	0
1	0	0	0	0
1	1	1	0	1







A . P . U
ASIA PACIFIC UNIVERSITY
OF TECHNOLOGY & INNOVATION

Inputs			Intermediates					Output
C	B	A	$A.B.C$	\overline{B}	\overline{C}	$\overline{B}+\overline{C}$	$A.(\overline{B}+\overline{C})$	Q
0	0	0	0	1	1	1	0	0
0	0	1	0	1	1	1	1	1
0	1	0	0	0	1	1	0	0
0	1	1	0	0	1	1	1	1
1	0	0	0	1	0	1	0	0
1	0	1	0	1	0	1	1	1
1	1	0	0	0	0	0	0	0
1	1	1	1	0	0	0	0	1

Boolean Algebra

- Digital computers contain circuits that implement Boolean functions.
- The **simpler** that we can make a Boolean function, the **smaller the circuit** that will result.
 - Simpler circuits are cheaper to build, consume less power, and run faster than complex circuits.
- With this in mind, we always want to **reduce our Boolean functions to their simplest form**.
- There are a number of **Boolean identities** that help us to do this.

Boolean Identities



A . P . U
ASIA PACIFIC UNIVERSITY
OF TECHNOLOGY & INNOVATION

Identity	Name
$(x')' = x$	Involution Law
$x + x' = 1$ $x \cdot x' = 0$	Complementarity
$x + x = x$ $x \cdot x = x$	Idempotent Laws
$x + 0 = x$ $x \cdot 1 = x$	Identity Laws
$x + 1 = 1$ $x \cdot 0 = 0$	Dominance Laws
$x + y = y + x$ $xy = yx$	Commutative Laws

Boolean Identities



A.P.U.
UNIVERSITY
& INNOVATION

Identity	Name
$x + (y + z) = (x + y) + z$ $x(yz) = (xy)z$	Associative Laws
$x + yz = (x + y)(x + z)$ $x(y + z) = xy + xz$	Distributive Laws
$(xy)' = x' + y'$ $(x + y)' = x'y'$	De Morgans' Laws
$x + (xy) = x$ $x(x + y) = x$	Absorption Laws
$x + x'y = x + y$ $x(x' + y) = xy$	Redundancy Laws
$xy + x'z + yz = xy + x'z$ $(x+y)(x'+z)(y+z) = (x+y)(x'+z)$	Consensus Laws

Example 1

$$A + A \cdot B = A$$

Proof Steps

$$\begin{aligned} & A + A \cdot B \\ &= A \cdot 1 + A \cdot B \\ &= A \cdot (1 + B) \\ &= A \cdot 1 \\ &= A \end{aligned}$$

(Absorption Theorem)

Justification

Identity element: $A \cdot 1 = A$

Distributive

$$1 + B = 1$$

Identity element

- Our primary reason for doing proofs is to learn:
 - Careful and efficient use of the identities and theorems of Boolean algebra, and
 - How to choose the appropriate identity or theorem to apply to make forward progress

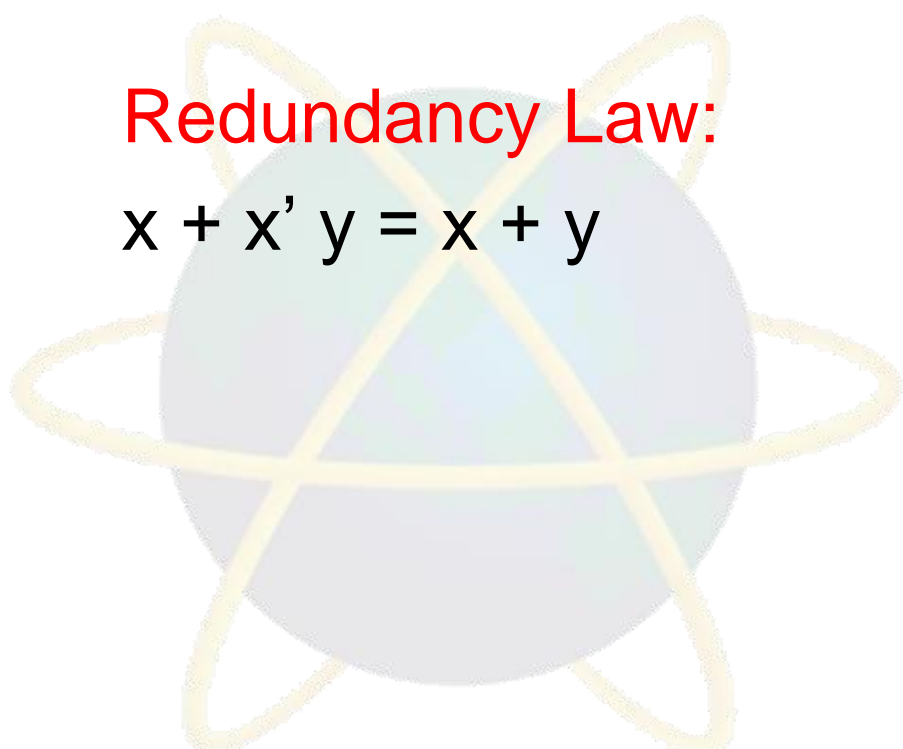
How about....

Distributive Law:

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

Redundancy Law:

$$x + x' y = x + y$$



Example 2

$$AB + \overline{A}C + BC = AB + \overline{A}C \text{ (Consensus Laws)}$$

Proof Steps

$$\begin{aligned} & AB + \overline{A}C + BC \\ &= AB + \overline{A}C + 1 \cdot BC \\ &= AB + \overline{A}C + (A + \overline{A}) \cdot BC \\ &= AB + \overline{A}C + ABC + \overline{A}BC \\ &= AB + ABC + \overline{A}C + \overline{A}CB \\ &= AB \cdot 1 + ABC + \overline{A}C \cdot 1 + \overline{A}CB \\ &= AB(1+C) + \overline{A}C(1+B) \\ &= AB \cdot 1 + \overline{A}C \cdot 1 \\ &= AB + \overline{A}C \end{aligned}$$

Justification

Identity element

Complement

Distributive

Commutative

Identity element

Distributive

$1+X = 1$

Identity element

Example 3

$$(A + B) (A + B') (AC')'$$

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

Involution & DeMorgan's Law

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

Distributive

$$= ((A + BB') + A(B + B')) (A' + C)$$

Cummutative & Distributive

$$= ((A + 0) + A(1)) (A' + C)$$

Complementarity

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

Idempotent

$$= AC$$

Redundancy

Use DeMorgan's Theorem:

1. Interchange AND and OR operators
2. Complement each constant and literal

Example 4

$$\begin{aligned} & \mathbf{AB + \bar{A}CD + \bar{A}BD + \bar{A}C\bar{D} + ABCD} \\ &= AB + ABCD + \bar{A}CD + \bar{A}C\bar{D} + \bar{A}BD \\ &= AB + AB(CD) + \bar{A}C(D + \bar{D}) + \bar{A}BD \\ &= AB + \bar{A}C + \bar{A}BD = B(A + \bar{A}D) + \bar{A}C \\ &= B(A + D) + \bar{A}C \end{aligned}$$

Boolean Identities

Simplify the following expression :

(a) $F = C + \overline{BC}$

(b) $F = \overline{AB}(\overline{A} + B)(\overline{B} + B)$

(c) $F = (A + C)(AD + A\overline{D}) + AC + C$

Minterms and Maxterms

- Any boolean expression may be expressed in terms of either minterms or maxterms.
- To do this we must first define the concept of a literal.
- A **literal** is a **single variable** within a term which **may or may not be complemented**. For an expression with N variables, minterms and maxterms are defined as follows :
 - A minterm is the **product of N distinct literals** where each literal occurs exactly once.
 - A maxterm is the **sum of N distinct literals** where each literal occurs exactly once.

Minterms & Maxterms for 2 variables

- Two variable minterms

x	y	Minterm
0	0	$\overline{x} \overline{y}$
0	1	$\overline{x} y$
1	0	$x \overline{y}$
1	1	$x y$

- The minterm is,
 - ‘1’ means the variable is “Not Complemented” and
 - ‘0’ means the variable is “Complemented”.

Minterms & Maxterms for 3 variables

x	y	z		Minterm	Maxterm
0	0	0		$\bar{x} \bar{y} \bar{z}$	$x + y + z$
0	0	1		$\bar{x} \bar{y} z$	$x + y + \bar{z}$
0	1	0		$\bar{x} y \bar{z}$	$x + \bar{y} + z$
0	1	1		$\bar{x} y z$	$x + \bar{y} + \bar{z}$
1	0	0		$x \bar{y} \bar{z}$	$\bar{x} + y + z$
1	0	1		$x \bar{y} z$	$\bar{x} + y + \bar{z}$
1	1	0		$x y \bar{z}$	$\bar{x} + \bar{y} + z$
1	1	1		$x y z$	$\bar{x} + \bar{y} + \bar{z}$

Maxterm is the complement of minterm

‘0’ means the variable is “Not Complemented” and

‘1’ means the variable is “Complemented”.

Representing Boolean Functions

- **SOP** and **POS**

- Boolean expressions can be manipulated into 2 forms

- Standardized forms are required for Boolean expressions to simplify communication of the expressions:

- **Sum-of-products (SOP)/MINTERM**

- Example: $F(A, B, C, D) = AB + \overline{B}C\overline{D} + AD$

- **Product-of-sums (POS)/MAXTERM**

- Example: $F(A, B, C, D) = (A + B)(\overline{B} + C + \overline{D})(A + D)$

Sum-Of-Minterm (SOP)

- Sum-Of-Minterm (SOP) canonical form:
Sum of minterms of entries that evaluate to '**1**'

x	y	z	F	Minterm
0	0	0	0	
0	0	1	1	$\bar{x} \bar{y} z$
0	1	0	0	
0	1	1	0	
1	0	0	0	
1	0	1	0	
1	1	0	1	$x y \bar{z}$
1	1	1	1	$x y z$

Focus on
the '**1**'
entries

$$\bar{x} \bar{y} z + x y \bar{z} + x y z$$

Product-Of-Maxterm (POS)

- Product-Of-Maxterm (POS) canonical form:
Product of maxterms of entries that evaluate to '**0**'

x	y	z	F	Maxterm
0	0	0	1	
0	0	1	1	
0	1	0	0	$(x + \bar{y} + z)$
0	1	1	1	
1	0	0	0	$(\bar{x} + y + z)$
1	0	1	1	
1	1	0	0	$(\bar{x} + \bar{y} + z)$
1	1	1	1	

Focus on
the '**0**'
entries

$$(x + \bar{y} + z) (\bar{x} + y + z) (\bar{x} + \bar{y} + z)$$

Example

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

- There are four 1s in the output and the corresponding binary value are 011, 100, 110, and 111.

$$011 \rightarrow \bar{A}BC$$

$$100 \rightarrow A\bar{B}\bar{C}$$

$$110 \rightarrow AB\bar{C}$$

$$111 \rightarrow ABC$$

$$SOP = \bar{A}BC + A\bar{B}\bar{C} + AB\bar{C} + ABC$$

- There are four 0s in the output and the corresponding binary value are 000, 001, 010, and 101.

$$000 \rightarrow A + B + C$$

$$001 \rightarrow A + B + \bar{C}$$

$$010 \rightarrow A + \bar{B} + C$$

$$101 \rightarrow \bar{A} + B + \bar{C}$$

$$POS = (A + B + C)(A + B + \bar{C})(A + \bar{B} + C)(\bar{A} + B + \bar{C})$$

Example

Find the sum of products expansion for the function $F(A, B, C) = (A + B)C'$ by using truth table

A	B	C	A+B	C'	(A+B)C'
1	1	1	1	0	0
1	1	0	1	1	1
1	0	1	1	0	0
1	0	0	1	1	1
0	1	1	1	0	0
0	1	0	1	1	1
0	0	1	0	0	0
0	0	0	0	1	0

Three 1s in the output. (110, 100, and 010).

$$SOP = ABC\bar{C} + A\bar{B}\bar{C} + \bar{A}B\bar{C}$$

Five 0s in the output (111, 101, 011, 001, and 000).

$$POS = (\bar{A} + \bar{B} + \bar{C})(\bar{A} + B + \bar{C})(A + \bar{B} + \bar{C})(A + B + \bar{C})(A + B + C)$$

Karnaugh Map

- Karnaugh Maps (K-Maps) are a graphical method of visualizing the 0's and 1's of a boolean function
 - K-Maps are very useful for performing Boolean minimization – SOP expansions.
- Will work on 2 and 3variable K-Maps in this class.
- Karnaugh maps can be easier to use than boolean equation minimization once you get used to it.

Karnaugh Map

- A K-map has a square for each '1' or '0' of a boolean function.
- Two variable K-map has $2^2 = 4$ squares
- Three variable K-map has $2^3 = 8$ squares
- Four variable K-map has $2^4 = 16$ squares

2 variable

3 variable

4 variable

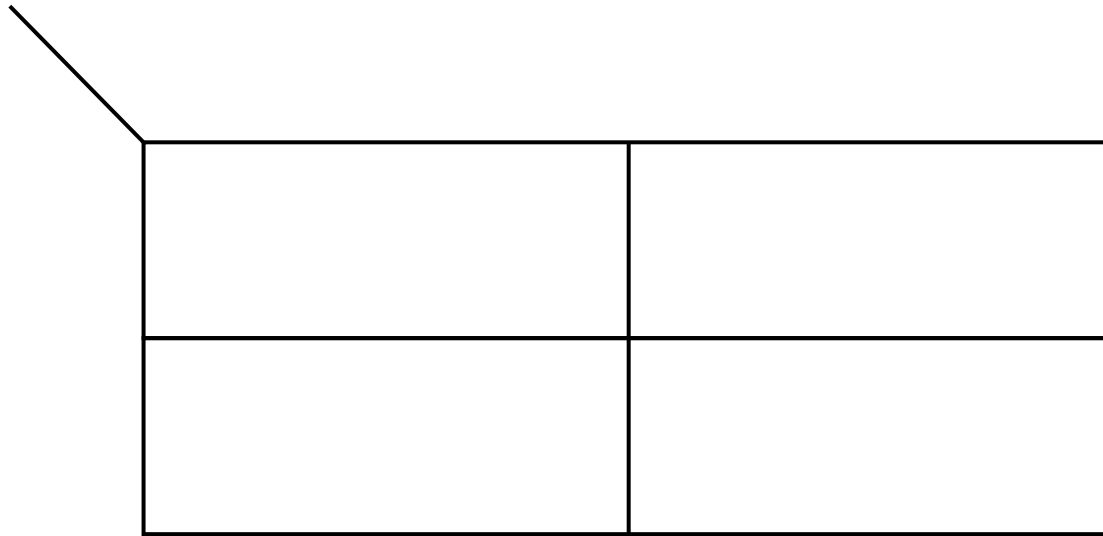
Karnaugh Map

- 1s is placed in the square representing a minterm if this minterm is present in the expansion.
 - The goal is to identify the largest blocks of 1s in the map and to cover all the 1s using the fewest blocks needed, using the largest blocks first.
- note : the largest possible blocks are always chosen, but we must **always choose a block if it is the only block 1s covering a 1s in a k-map**
- Must group 1s in either 1, 2, 4, 8, or 16 and must be adjacent.
 - Eliminate the variable that have value false and true to get the simple expression.

Plotting Functions on K-Maps

✖ Two-variable map

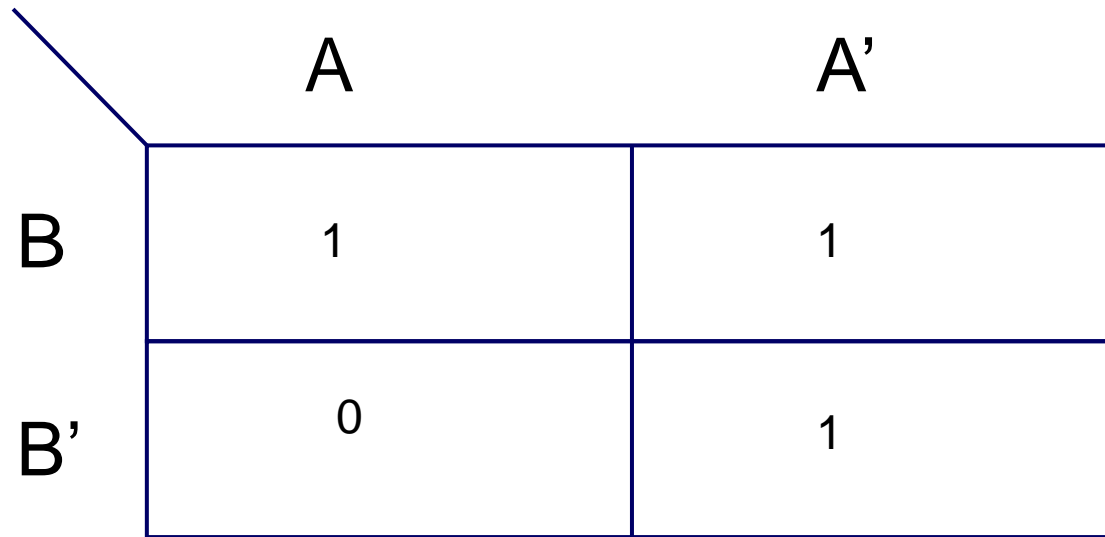
Ex: $A'B' + A'B + AB$



Plotting Functions on K-Maps

✖ Two-variable map

Ex: $A'B' + A'B + AB$



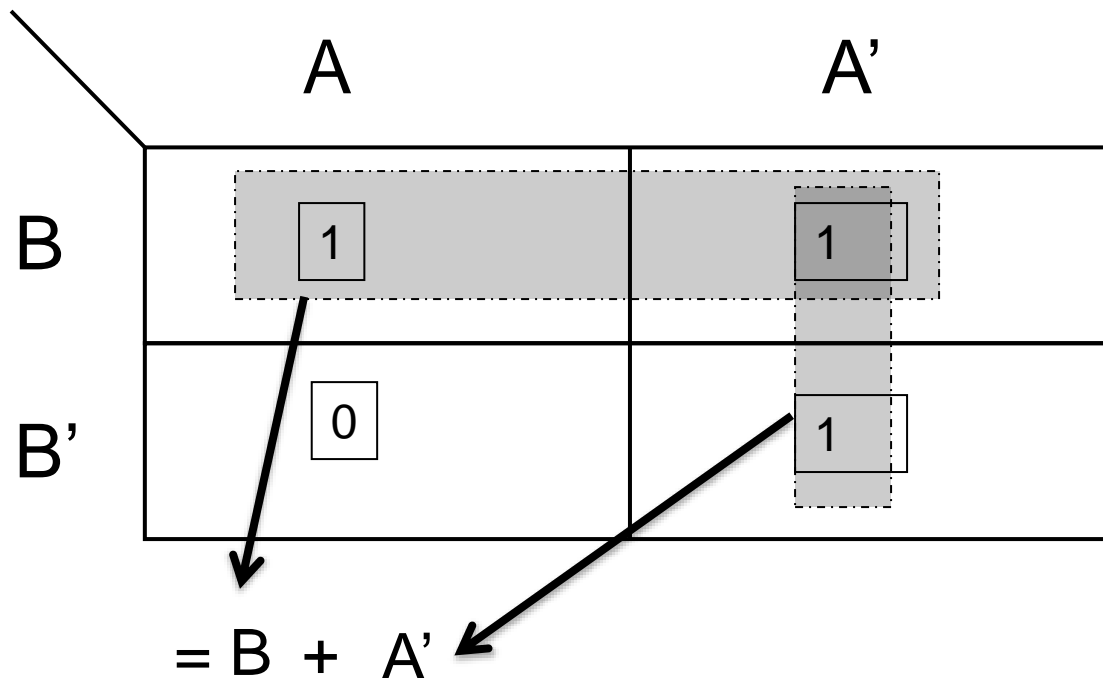
	A	A'
B	1	1
B'	0	1

Plotting Functions on K-Maps

- **Two-variable map**

Ex: $A'B' + A'B + AB$

On a two-variable map, look for a pair of 1's that are either in the same row or column. If we find such a pair, we record the common variable.



Rule: 1

- A group of **four will have one common variable**. A pair must have **two common variables**.
- In the examples below, the common variable for the left map is C' and the common variable for the right group is B .

	$\sim A \sim B$	$\sim AB$	AB	$A \sim B$
$\sim C$	1	1	1	1
C				

	$\sim A \sim B$	$\sim AB$	AB	$A \sim B$
$\sim C$		1	1	
C		1	1	

Rule: 2

- In the examples below, the **left pair** has the **common variables BC**, while the **right pair** has the common variables **AB**'.
- A **solo / single 1** must have all 3 variables.

		~A~B	~AB	AB	A~B
~C					
C		<u>1</u>	1		

		~A~B	~AB	AB	A~B
~C					<u>1</u>
C					1

Rule: 3

- The map "wraps around" from the left edge to the right
- The common variables for the pair on the left are $B'C'$, while the group of four on the right has the common variable B' .

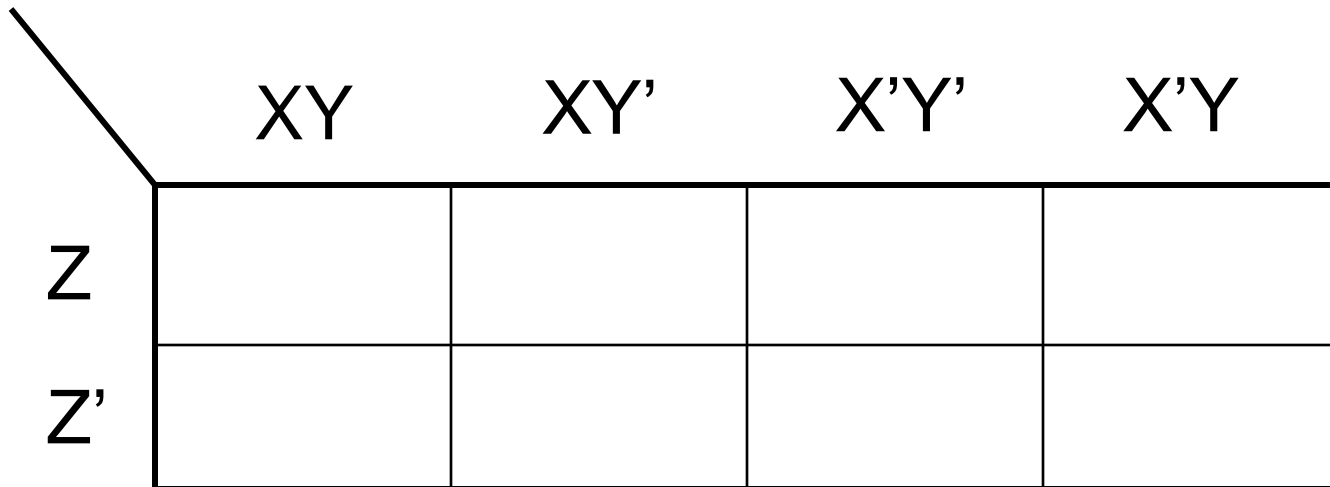
	$\sim A \sim B$	$\sim AB$	AB	$A \sim B$
$\sim C$	1			1
C				

	$\sim A \sim B$	$\sim AB$	AB	$A \sim B$
$\sim C$	1			1
C	1			1

Plotting Functions on K-Maps

- **Three-variable map**

Ex: $X'Y'Z' + X'YZ + XYZ' + XY'Z + XYZ + X'YZ'$



	XY	XY'	X'Y'	X'Y
Z				
Z'				

Plotting Functions on K-Maps

- **Three-variable map**

Ex: $X'Y'Z' + X'YZ + XYZ' + XY'Z + XYZ + X'YZ'$

	XY	XY'	X'Y'	X'Y
Z	1	1		1
Z'	1		1	1

Plotting Functions on K-Maps

- **Three-variable map**

Ex: $X'Y'Z' + X'YZ + XYZ' + XY'Z + XYZ + X'YZ'$

	XY	XY'	X'Y'	X'Y
Z	1	1		1
Z'	1		1	1

$$= Y + XZ + X'Z'$$

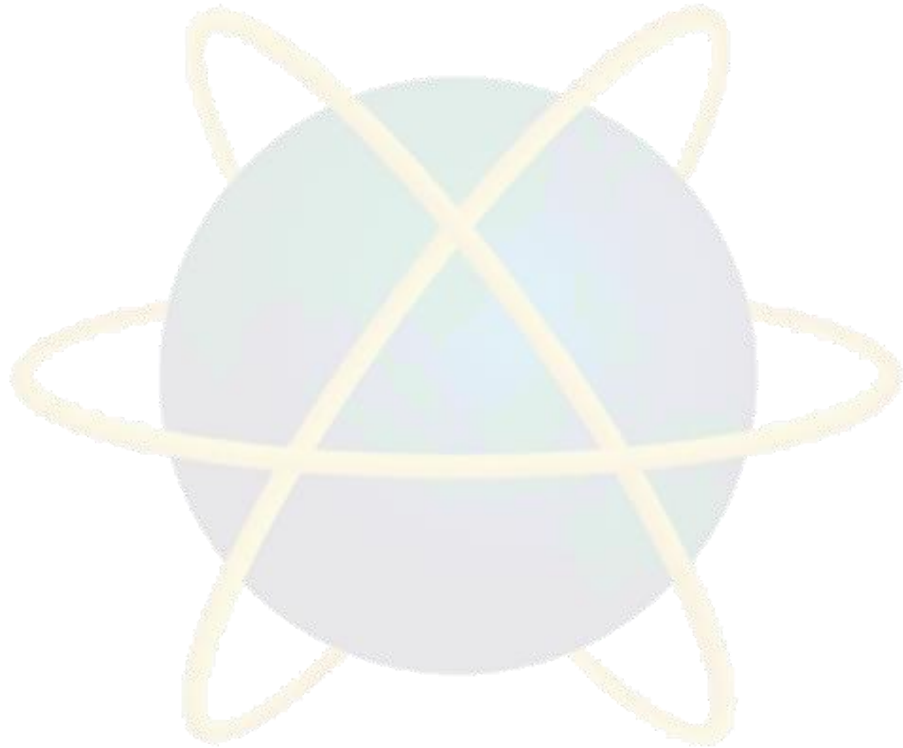
Example

Obtain the simplest form for the following minterm expression.

	$\overline{B}.\overline{C}$	$\overline{B}.C$	$B.C$	$B.\overline{C}$
\overline{A}	1	1 1	1 1	1
A	1	1 1	1 1	1 1

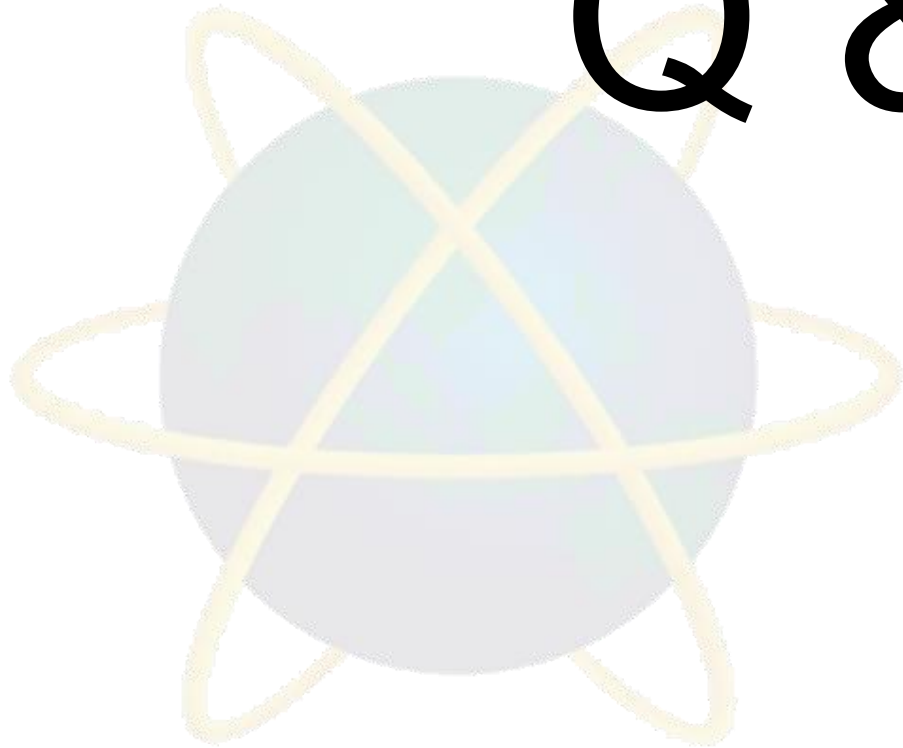
Quick Review Question

?



Question and Answer Session

Q & A



Next Session

Graph Theory

