Topic support guide

Cambridge International AS & A Level

Cambridge International AS and A Level Computer Science

9608

For examination from 2017

Topic 3.3.2 Boolean algebra

Cambridge Advanced

CAMBRIDGE
International Examinations

Cambridge International Examinations retains the copyright on all its publications. Registered Centres are permitted to copy material from this booklet for their own internal use. However, we cannot give permission to Centres to photocopy any material that is acknowledged to a third party even for internal use within a Centre.

© Cambridge International Examinations 2016
Version 1

A DIVISION OF
CAMBRIDGE ASSESSMENT

Contents

Introduction

How to use this guide

The aim of this guide is to facilitate your teaching of the Cambridge International AS and A Level Computer Science topic 3.3.2 Boolean algebra, part of the advanced theory topic 3.3 Hardware. The guidance and activities in this resource are designed to help teachers devise programmes of study which provide teaching time devoted to theory work as well as activities that consolidate learning.

Section 1 lists some key terms used in this topic and their definitions. Section 2 introduces the theory of Boolean algebra and provides some example expressions and logic circuits. Section 3 lists some online resources that you or your learners may find useful. Section 4 gives ideas for class and homework activities.

Learning objectives

Using this document should help you guide learners in the following syllabus learning objectives:

- show understanding of Boolean algebra
- show understanding of De Morgan's Laws
- perform Boolean algebra using De Morgan's Laws
- simplify a logic circuit/expression using Boolean algebra.

Prior knowledge

Before you begin teaching this topic you should:

- understand what Boolean algebra is
- be familiar with how to write Boolean expressions
- be familiar with the rules and laws that can be used to simplify Boolean expressions
- practise simplifying logic circuits and expressions.

1. Key terms

Word/phrase	Meaning
Boolean value	a value that is true or false
logic gate	a component in an electronic circuit that produces an output from a combination of inputs, based on a logical function such as AND, OR, or NOT
proposition	a statement or hypothesis
truth table	a table to show all the outputs from a combination of inputs

2. Theory and examples

2.1 What is Boolean algebra?

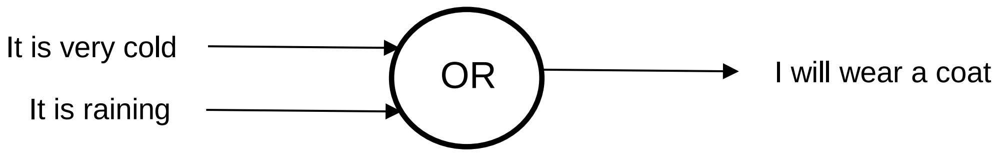
Boolean algebra is a form of mathematics that deals with statements and their Boolean values. It is named after its inventor George Boole, who is thought to be one of the founders of computer science. In Boolean algebra variables and functions take on one of two values: true or false. These values can also be referred to as 1 for true and 0 for false.

2.1.1 Example statements

If we take a simple statement we can start to see the operations of Boolean algebra:

> "I will wear a coat if it is very cold or if it is raining."

The **proposition** 'I will wear a coat' will be true if either of the propositions 'It is very cold' or 'It is raining' are true. We can represent this as a diagram:



As a **truth table** – a table that shows the output for each possible input combination – this is:

It is very cold	**It is raining**	**I will wear a coat**
False	False	False
False	True	True
True	False	True
True	True	True

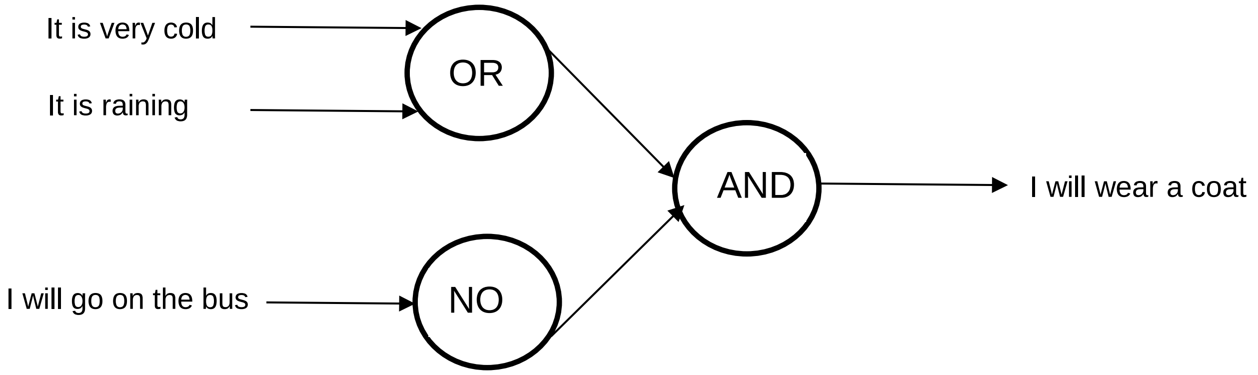
or

It is very cold	**It is raining**	**I will wear a coat**
0	0	0
0	1	1
1	0	1
1	1	1

We could then make our statement more complex by saying:

> "If I do not go on the bus, I will wear a coat if it is very cold or if it is raining."

We can represent this as a diagram:



If we look at the logic of our diagram we can see that if it is very cold or it is raining and I am not going on the bus, I will wear a coat. But if I go on the bus I will not wear a coat regardless of whether it is very cold or it is raining.

2.2 Writing Boolean expressions

2.2.1 Brackets

Boolean operations have an order in which they are carried out. This order is:

- Highest – NOT
- Middle – AND
- Lowest – OR

However, anything that appears inside brackets has the normal mathematical rule apply, that it is always carried out first.

We can begin to create a Boolean expression from our diagram. Note that the brackets around the OR statement mean that this will be evaluated before the AND statement:

Coat = ((NOT Bus) AND (Very cold OR Raining))

2.2.2 Symbols

Boolean expressions often use symbols for the operators. Different symbols can be used, but we will use:

Symbol	Operator
$\overline{}$	NOT
·	AND
+	OR

In Boolean algebra, parts of a statement are often given a single letter. This means we can write our statement as:

$$C = ((\overline{B}) \cdot (V + R))$$

If we now consider the order that Boolean operators have to be carried out in, the NOT statement will be evaluated first, so we can simplify our statement slightly by removing the brackets:

$$C = \overline{B} \cdot (V + R)$$

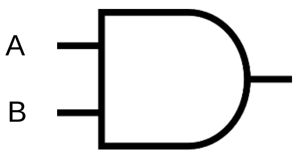
2.3 Boolean laws and simplification

2.3.1 Boolean laws

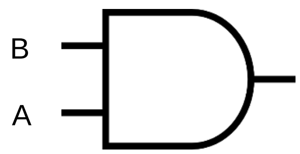
There are some further general rules (laws) in Boolean algebra. We will use the values A, B and C to demonstrate these rules. The rules apply to the AND and OR logic gates:

Rule (law)	AND	OR
Commutative	A · B = B · A	A + B = B + A
Associative	(A · B) · C = A · (B · C)	(A + B) + C = A + (B + C)
Distributive	A · (B + C) = A · B + A.C	A + (B · C) = (A + B) · (A + C)

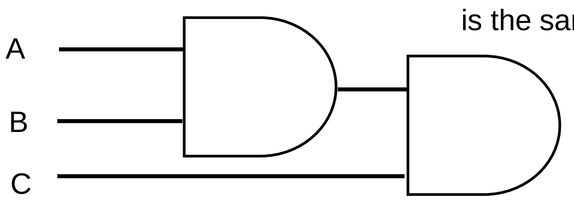
Looking at the table above, we can see in commutative law that A.B is the same as B.A. If we represent this in terms of a logic gate we are saying that:



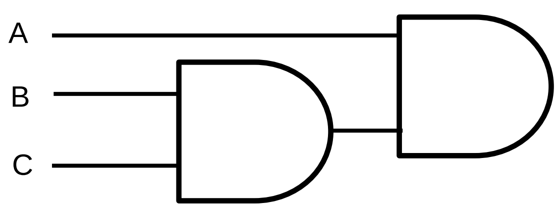
is the same as



We can also see in associative law that (A.B).C is the same as A.(B.C). If we represent this in terms of a logic circuit we can say that:



is the same as



2.3.2 Simplification (reduction) rules

There are also some simplifications we need to be aware of for Boolean operators:

$$A \cdot A = A$$

$$A + A = A$$

$(\overline{\overline{A}}) = A$

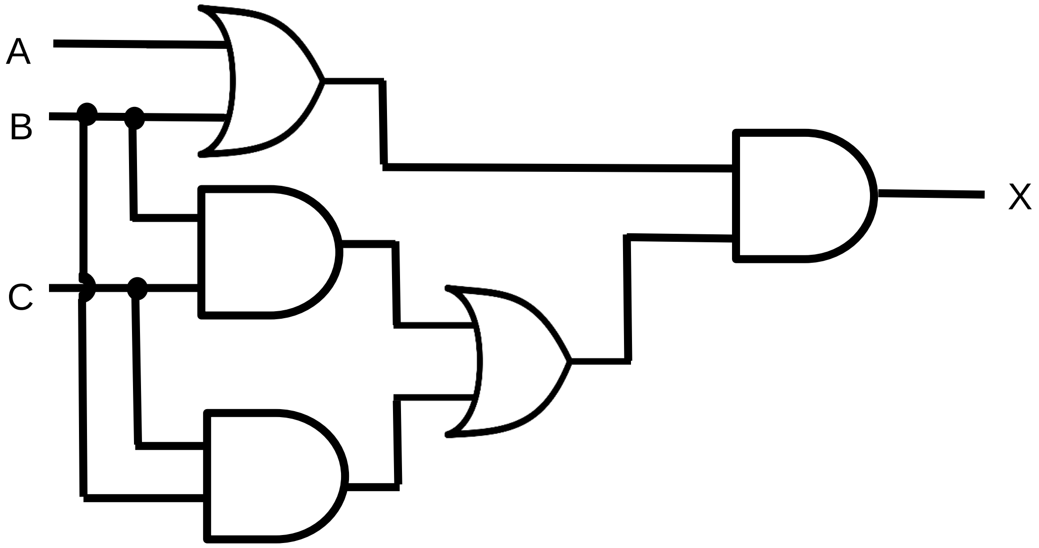
$A \cdot \overline{A} = 0$

$A + \overline{A} = 1$

By looking at our example and from understanding the rules that are applied and the simplifications that can occur, we now have a greater understanding of the workings of Boolean algebra.

2.3.3 Worked simplification example

We can now put our knowledge into practice with a further example. Consider the following logic circuit:



1) The first set of gates gives:

A + B
B · C
C · B

2) The output of the two AND gates is passed through an OR gate:

B · C + C · B

3) The output of this and the OR gate (A + B) is then passed through an AND gate. The whole expression is therefore:

X = (A + B) · ((B · C) + (C · B))

4) We can now apply the simplification rule A + A = A to simplify the expression:

(B · C) + (C · B) = (B · C)

5) The final simplified expression is therefore:

X = (A + B) · (B · C)

We now have a simpler expression than we first started off with and as a result we can create a simpler logic circuit.

2.4 De Morgan's Laws

2.4.1 What are De Morgan's Laws?

Augustus De Morgan was a contemporary of George Boole. He was the first professor of mathematics at the University of London. De Morgan did not actually create the law given his name, but is credited with stating it.

De Morgan's Laws are that:

1) when the NOT operator is applied to the AND of two variables, it is equal to the NOT applied to each of the variables with an OR in between:

$(\overline{A \cdot B}) = \overline{A} + \overline{B}$

2) when the NOT operator is applied to the OR of two variables, it is equal to the NOT applied to each of the two variables with an AND in between:

$(\overline{A + B}) = \overline{A} \cdot \overline{B}$

2.4.2 Worked example

We can apply De Morgan's Laws to the statement we made:

> "If I do not go on the bus, I will wear a coat if it is very cold or if it is raining."

We can simplify this statement to show the outcome 'I am not going to wear a coat':

Original statement: $C = \overline{B} \cdot (V + R)$

We can add NOT to show I am not going to wear a coat:

$\overline{C} = \overline{(\overline{B} \cdot (V + R))}$

We can then apply De Morgan's Law:

$\overline{C} = (\overline{\overline{B}}) + \overline{(V + R)}$

Then simplify and apply De Morgan's Law again, to give:

$\overline{C} = B + \overline{V} \cdot \overline{R}$

3. Online resources

The following are useful resources for understanding Boolean algebra.

The content of websites is dynamic and constantly changing. Schools are strongly advised to check each site for content and accessibility prior to using it with learners. Cambridge International Examinations is not responsible for the accuracy or content of information contained in these sites. The inclusion of a link to an external website should not be understood to be an endorsement of that website or the site's owners (or their products/services).

3.1 Websites

Learn about electronics website. Explains the laws and rules for simplification of Boolean expressions, as well as De Morgan's laws, and includes a worked example:

www.learnabout-electronics.org/Digital/dig23.php

Worked examples of simplification of Boolean expressions, from Mississippi College Computer Science site:

http://sandbox.mc.edu/~bennet/cs110/boolalg/simple.html

WJEC documentation of Boolean algebra simplification that includes worked examples, exercises and solutions for simplification of circuits and expressions:

www.wjec.co.uk/uploads/publications/4891.doc?language_id=1

4. Class and homework activities

4.1 Quiz

Multiple-choice items on Boolean algebra and other aspects of digital logic on the Learn about electronics site:

www.learnabout-electronics.org/Digital/dig25.php

Boolean Algebra quiz from the University of Surrey – most questions can be used:

www.ee.surrey.ac.uk/Projects/Labview/boolalgebra/quiz/index.html

4.2 Homework questions

1. Use Boolean algebra to simplify the following expression:

   $(\mathrm{A} \cdot \overline{\mathrm{A}}) + \mathrm{B}$

2. (i) Use Boolean algebra to simplify the following expression:

   $(\mathrm{A} \cdot \mathrm{B}) + (\overline{\mathrm{A}} \cdot \mathrm{B})$

   (ii) Draw a logic circuit that corresponds to your simplified expression

3. Use Boolean algebra to simplify the following expression:

   $\mathrm{A} \cdot \mathrm{B} + \mathrm{A} \cdot (\mathrm{B} + \mathrm{C})$