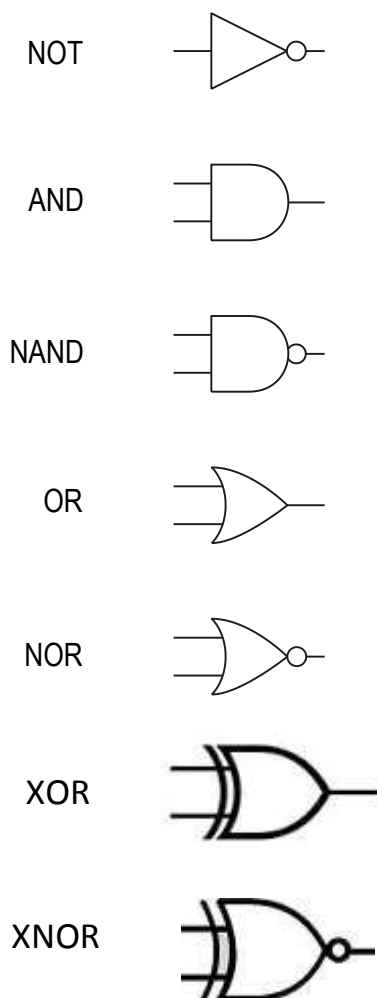


BASIC LOGIC GATES

There are seven different logic gates; these are the NOT, AND, OR, NAND, NOR, XOR and the XNOR.

When drawing circuits containing logic gates it is common to use logic symbols.



('X' stands for exclusive)

TRUTH TABLES

The easiest way to represent how each gate behaves is to make use of Truth Tables.

A Truth Table shows all possible combinations of inputs and outputs to a logic gate.

Electronics is concerned with the processing of electrical signals.



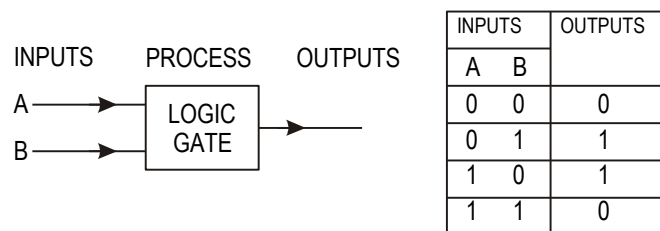
Input signals come from a variety of sources - a switch from a keyboard; a bar code reader; a temperature sensor; another part of a computer.

Output signals can have a variety of destinations - a monitor; a modem; an alarm; another part of a computer.

Digital signals can be at a HIGH voltage level or a LOW voltage level.

In logic circuits a LOW signal is said to be at logic '0' a HIGH signal at logic '1'.

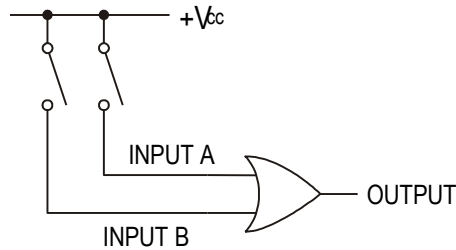
The results can be recorded and used in a number of formats, the most common being shown below.



Results displayed in this way are known as TRUTH TABLES.

STRUCTURE AND LAYOUT OF TRUTH TABLES

Consider the inputs to a logic gate as two switches.



It is possible for each switch to be in one of two positions, either, on (1) or off (0). These positions are known as INPUT STATES.

For two inputs, there are only 4 possible combinations of input states:

A-off & B-off; A-off & B-on; A-on & B-off; A-on & B-on.

Input A	Input B
0	0
0	1
1	0
1	1

For three inputs, there are 8 possible combinations.

In general, for **n** inputs there are **2ⁿ** possible combinations of input states.

ASSIGNMENT 1

1. How many combinations of input states would there be for a 6 input system?
2. Write down the 8 possible combinations of input states for a 3 input system.

TRUTH TABLES FOR INDIVIDUAL LOGIC GATES

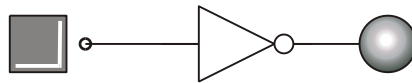
CIRCUIT SIMULATION SOFTWARE

It is possible to use circuit simulation software such as 'Yenka' to investigate electric and electronic circuits.

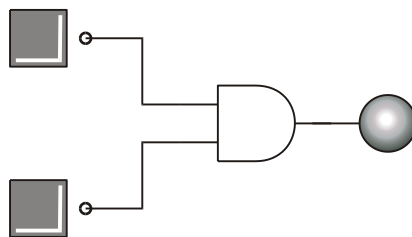
Use 'Yenka' to determine the truth table for each of the following gates

Use latching logic inputs and a logic indicator at the output.

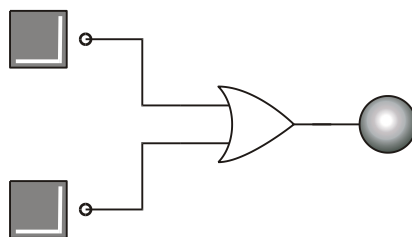
1. NOT gate



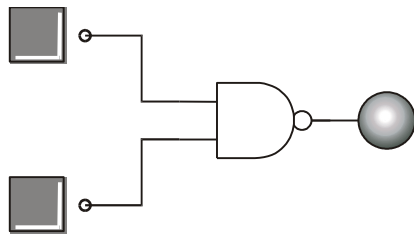
2. AND gate



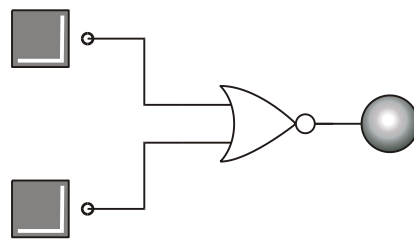
3. OR gate



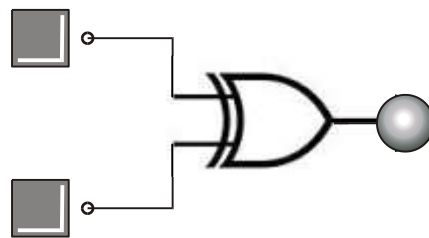
4. NAND gate



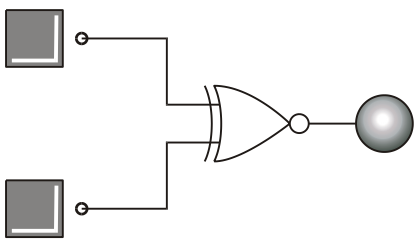
5. NOR gate



6. XOR gate



7. XNOR gate



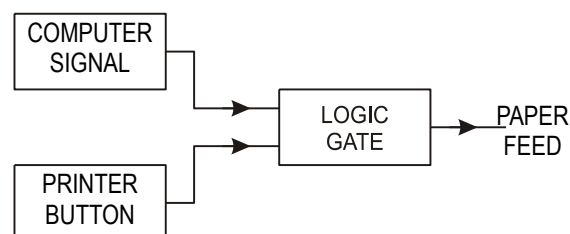
ASSIGNMENT 2

1. When is the output of an OR gate high?
2. When is the output of an AND gate high?
3. When is the output from an XOR gate high?
4. Why is the NOT gate sometimes called an 'INVERTER'?
5. The truth table below shows the output conditions for the various combinations of input conditions for AND, NAND, OR and NOR gates.

Inputs	AND	NAND	OR	NOR
0 0	0	1	0	1
0 1	0	1	1	0
1 0	0	1	1	0
1 1	1	0	1	0

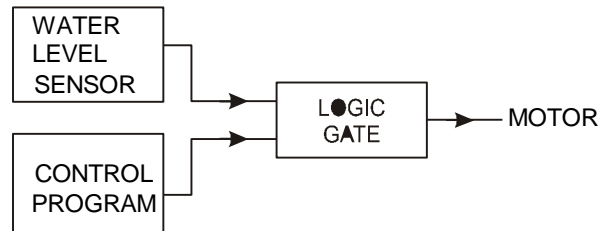
How does the output of the NAND gate compare with the output of the AND (and the output of the NOR compare with the OR)

6. The paper can be fed through a computer printer either by pressing the button on the printer (line feed) or by sending a signal from the computer.



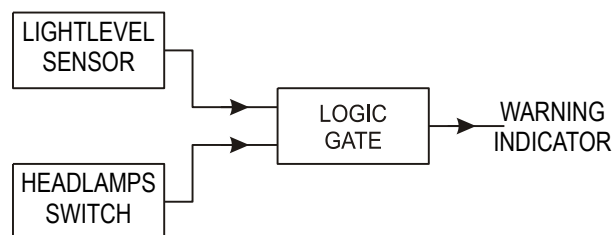
Which logic gate should be used for this operation?

7. The motor in a washing machine should not operate until a high signal is sent from the control program and the water level in the drum is high enough.



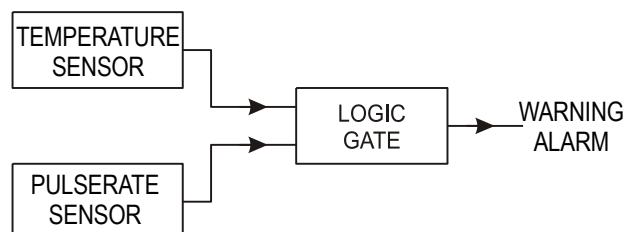
Which logic gate should be used for this operation?

8. To avoid accidents at times of poor visibility, a warning indicator in a car operates if the light level is too low (logic level 0) and the headlamps are switched off.



Which logic gate should be used for this operation?

9. In the maternity unit of a hospital, the temperature and pulse rate of premature babies has to be continually monitored. A warning alarm should sound if either the temperature or the pulse rate of the baby falls too LOW.



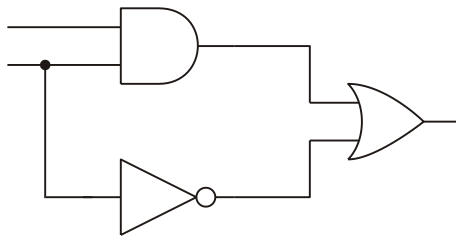
Which logic gate should be used for this operation?

COMBINATIONAL LOGIC SYSTEMS

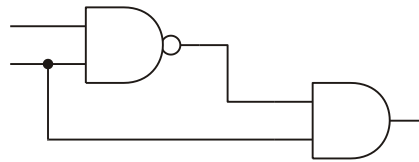
ASSIGNMENT 3

Complete a truth table for each of the combinations of gates shown below.

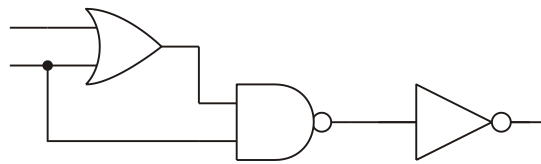
1.



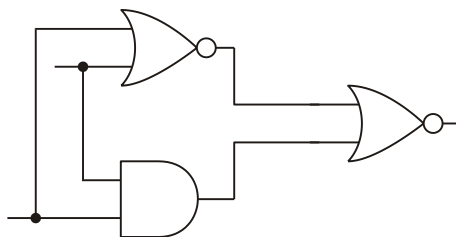
2.



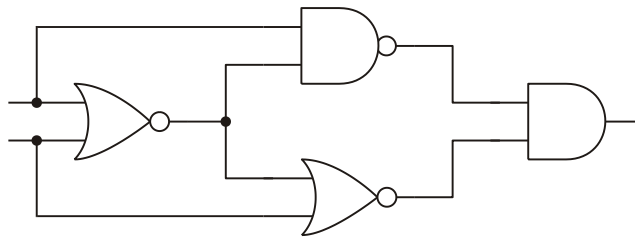
3.



4.



5.



Check your results on Yenka.

METHODS OF MAKING SPECIFIC GATES FROM NANDS

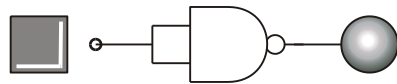
CIRCUIT SIMULATION SOFTWARE

Use Yenka to determine the truth table for each of the following network of NAND gates.

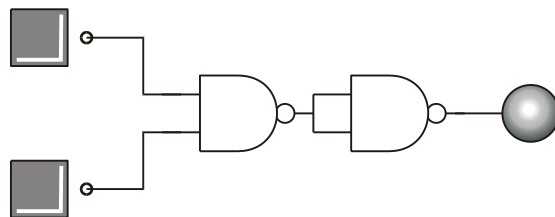
Compare the truth table you obtain with truth tables for the individual gates and decide which gate is the equivalent to the NAND network.

In some of the networks the two inputs of the NAND gate have been connected together to make a single input.

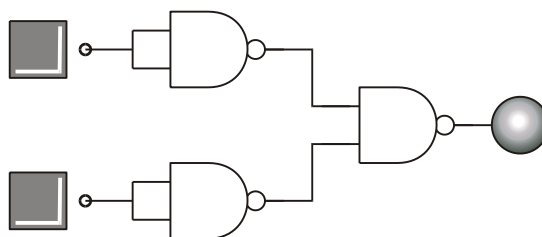
1.



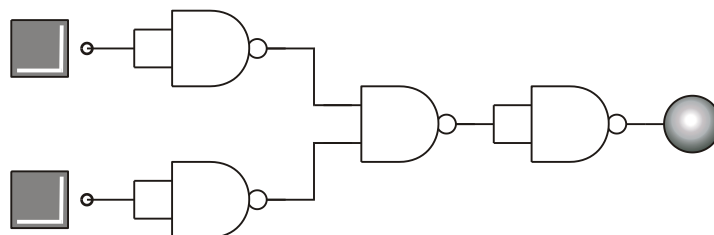
2.



3.



4.

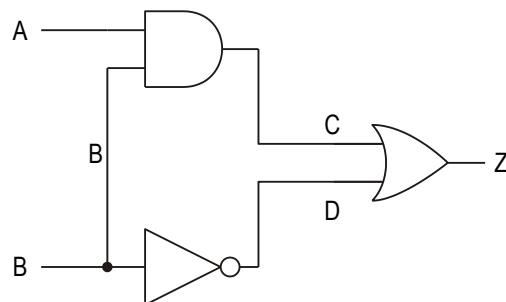


SIMPLIFICATION OF COMBINATIONAL LOGIC CIRCUITS

As has previously been stated it is possible to make all logic circuits from NAND gates only.

This section will examine a method for converting circuits that contain a number of different types of gates into one that uses NAND gates only.

Consider the circuit shown.

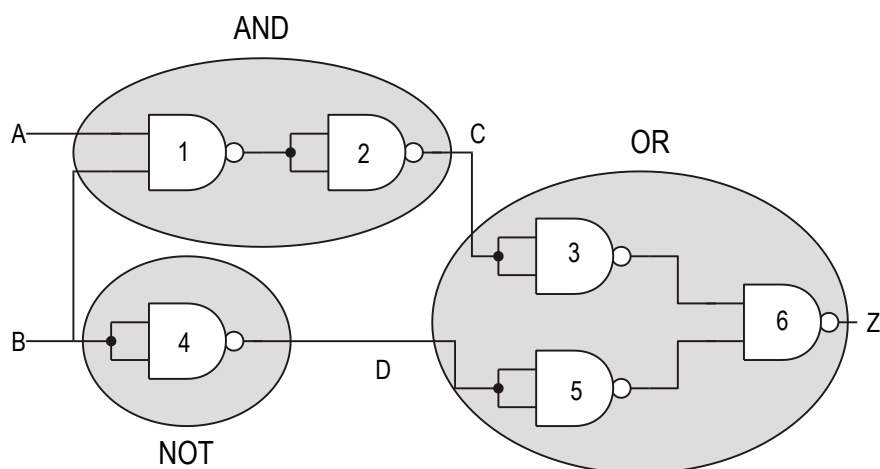


The system is made from an AND gate an OR gate and a NOT gate.

The problem is to design a system with the same Truth Table, but made from NAND gates only.

STEP 1

Redraw the circuit, replacing each gate with its NAND gate equivalent.

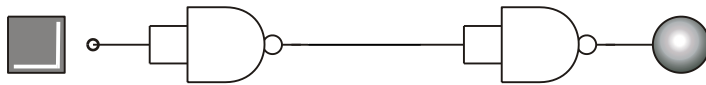


STEP 2

Examine the new arrangement and look for adjacent pairs of NOT gates.

In this circuit there are two such pairs. (2 & 3 and 4 & 5 are adjacent pairs)

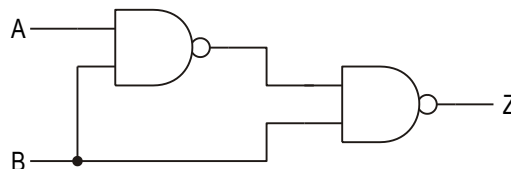
If you consider what happens when you feed a signal to a NOT gate then pass the signal on to another NOT gate you will find that the signal has been 'DOUBLE INVERTED' this in fact means that whenever you feed a signal to a pair of NOT gates you will get the same signal out.



Therefore pairs of NOT gates in series can be removed from the system without any effect.

STEP 3

Redraw the circuit with the NOT gates removed.



The final circuit has only two gates whereas the original circuit started with three gates, each of a different type. There are obvious implications in terms of cost for manufacturers if they are able to reduce logic circuits to situations where there are fewer gates as well as enabling them to use one type of gate. The original circuit would have required three IC's and the final circuit would only require one IC.

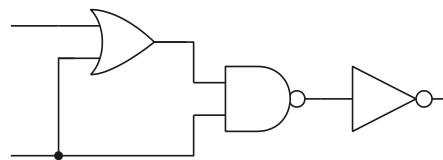
This method may not reduce the number of gates used on every occasion but it should reduce the number of IC's used.

This method is not very elegant and can be very demanding in terms of paper use and does not always lead to a very efficient use of NAND gates. The next section on Boolean algebra should allow us to design circuits more effectively and use fewer gates.

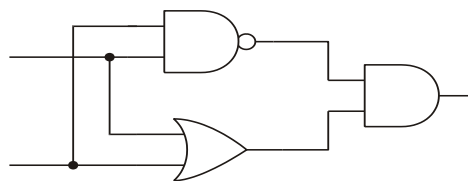
ASSIGNMENT 4

The following logic diagrams are constructed from basic gates. Using the method shown, construct equivalent circuits using NAND gates only.

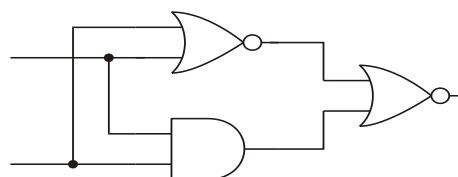
1.



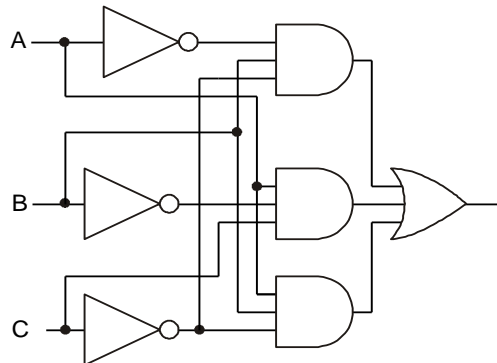
2.



3.



4. a) Construct a truth table for the logic circuit shown.



b) Redraw the circuit using NAND gates only.

c) Simplify the NAND circuit.

d) Construct a truth table for the finished NAND circuit.

BOOLEAN ALGEBRA

Boolean algebra is a special form of algebra that has been developed for binary systems. It was developed by George Boolean in 1854 and can be very useful for simplifying and designing logic circuits.

VARIABLES:

The most commonly used variables in logic circuit design are capital letters; such as A, B, C, Z and so on and are used to annotate inputs and outputs to systems.

In digital electronics we consider situations where the variables can only have one of two possible values, i.e. 'Logical 0' or 'Logical 1'.

The statement $A = 1$ means that the variable A has the value of Logic 1. Similarly, if $B = 0$ it means that variable B has the value of logic 0.

Logical Operations: In Boolean algebra there are three logical operators, these are the AND operation, the OR operation and the Inversion.

AND OPERATOR

The AND operation can be represented in Boolean notation by

$$A \bullet B = Z$$

The dot between the A and the B is read as AND.

OR OPERATOR

The OR operation can be represented in Boolean notation by

$$A+B = Z$$

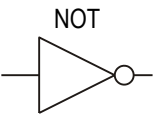
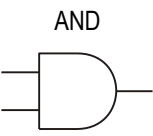
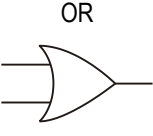
The + between the A and the B is read as OR.

INVERSION OPERATOR

The statement $\bar{A} = Z$ means that Z is not equal to A.

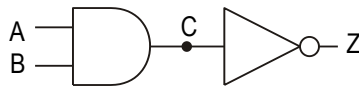
The variable is read as A bar and usually means NOT A. The bar over the top of the variable changes its value, or inverts it. This is known as the NOT operation.

BASIC LOGIC GATES AND THEIR BOOLEAN REPRESENTATIONS

LOGIC SYMBOL	BOOLEAN EXPRESSION	DESCRIPTION	TRUTH TABLE															
<p>NOT</p> 	$\bar{A} = Z$	NOT A EQUALS Z	<table border="1"><thead><tr><th>A</th><th>Z</th></tr></thead><tbody><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></tbody></table>	A	Z	0	1	1	0									
A	Z																	
0	1																	
1	0																	
<p>AND</p> 	$A \cdot B = Z$	A AND B EQUALS Z	<table border="1"><thead><tr><th>A</th><th>B</th><th>Z</th></tr></thead><tbody><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></tbody></table>	A	B	Z	0	0	0	0	1	0	1	0	0	1	1	1
A	B	Z																
0	0	0																
0	1	0																
1	0	0																
1	1	1																
<p>OR</p> 	$A+B = Z$	A OR B EQUALS Z	<table border="1"><thead><tr><th>A</th><th>B</th><th>Z</th></tr></thead><tbody><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></tbody></table>	A	B	Z	0	0	0	0	1	1	1	0	1	1	1	1
A	B	Z																
0	0	0																
0	1	1																
1	0	1																
1	1	1																

NAND GATE

The NAND gate is made up from a combination of an AND gate followed by a NOT gate.



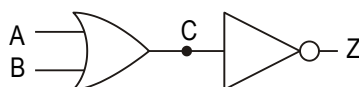
The signal at point C would be $A \bullet B$. This signal is then inverted by the NOT gate to give

$$\overline{A \bullet B} = Z$$

This reads as output Z is equal to A AND B all NOT

NOR GATE

The NOR gate is made up from a combination of an OR gate followed by a NOT.



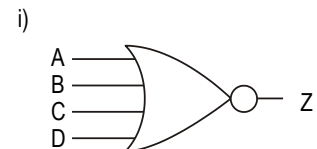
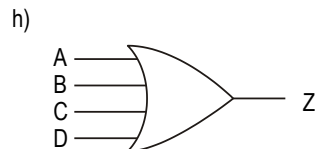
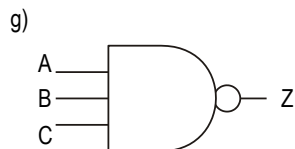
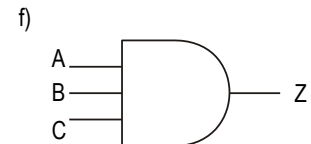
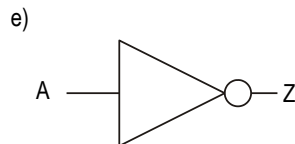
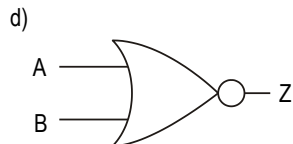
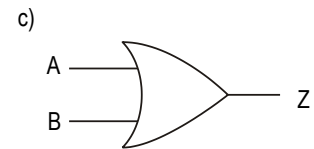
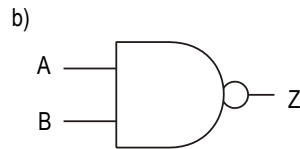
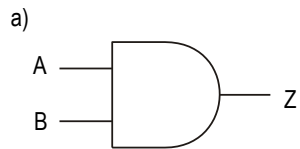
The signal at point C would be $A + B$. This signal is then inverted by the NOT gate to give

$$\overline{A + B} = Z$$

This reads as output Z is equal to A OR B all NOT

ASSIGNMENT 5

Write down the Boolean expression for each of the following logic gates.



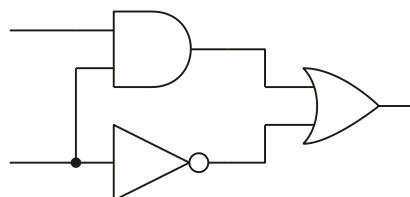
LAWS OF BOOLEAN ALGEBRA

The following is a summary of the basic laws of Boolean algebra.

- + represents logical operator OR
- represents Logical operator AND
- \bar{A} represents A bar i.e. NOT A (the inverse of A)

DERIVING THE BOOLEAN EXPRESSION FOR A CIRCUIT

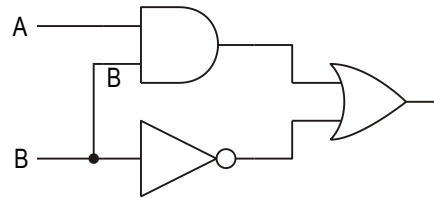
Consider the following circuit.



The Boolean expression for the circuit can be derived as follows:

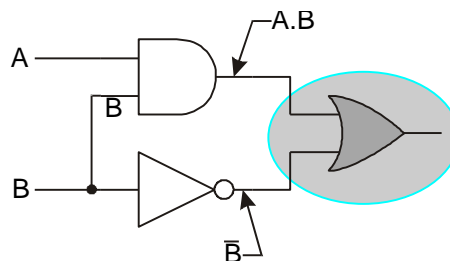
STEP 1

Label the inputs on the left-hand side of the diagram.



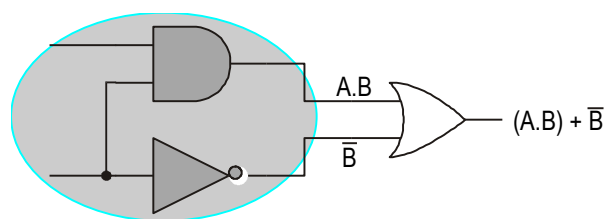
STEP 2

Consider each gate in turn. Use the Boolean notation to give the output of each gate in terms of its input. Write on the appropriate expression after each gate.



STEP 3

When outputs from other gates are inputs to a further gate, treat the expressions as you would any other equation and make use of brackets. Then write on the appropriate expression after the next gate and so on until you reach the final output.



STEP 4

Write down the final Boolean expression for the network.

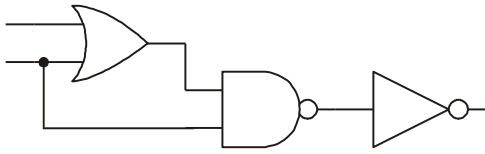
$$(A \bullet B) + \bar{B} = Z$$

Since normal rules of algebra hold, $\bar{B} + (A \bullet B) = Z$ would also be correct.

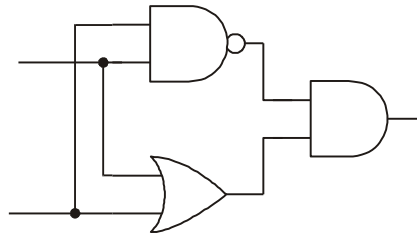
ASSIGNMENT 6 A

1. Derive the Boolean expression for each of the following circuits:

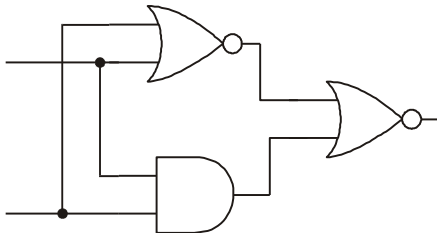
a)



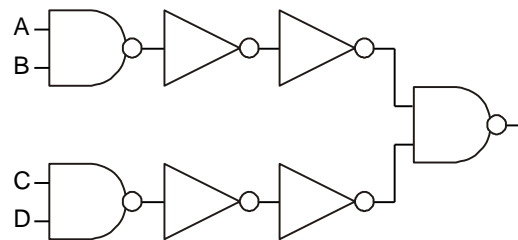
b)



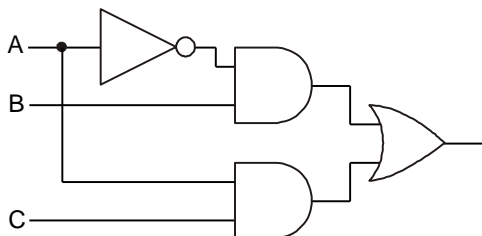
c)



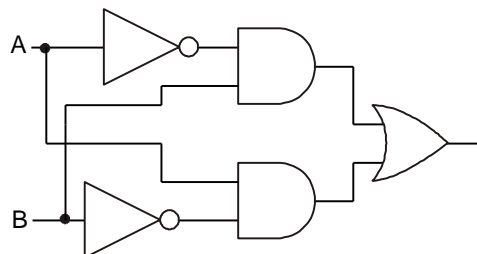
d)



e)



f)



2. Draw a gating arrangement to illustrate $Z = (A \bullet B) + C$

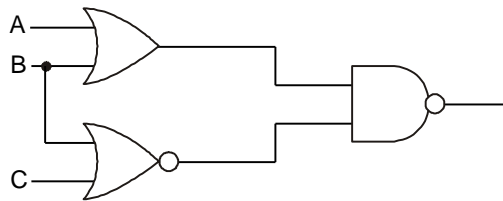
3. Draw a logic diagram to yield $D = B + C$

Develop it to obtain $Z = A \bullet D$

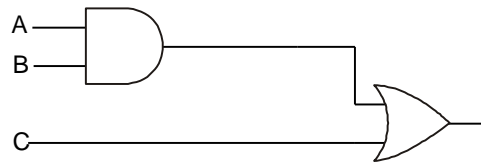
Write a Boolean equation for the overall behaviour.

4. Derive the Boolean equation and the truth table the following arrangements:

a)



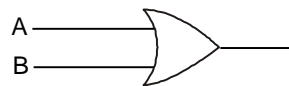
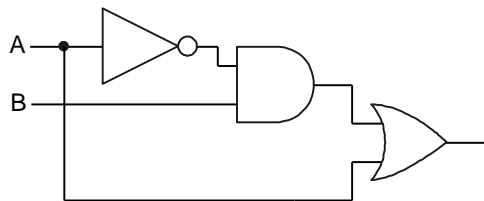
b)



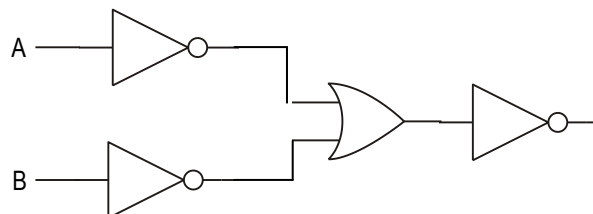
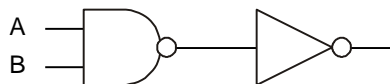
5. For each pair of circuits shown below:

- (i) Write a Boolean expression for each of these circuits;
- (ii) By constructing a truth table for each of them, show that they are equivalent;
- (iii) Draw the equivalent arrangements using only 2-input NAND gates.

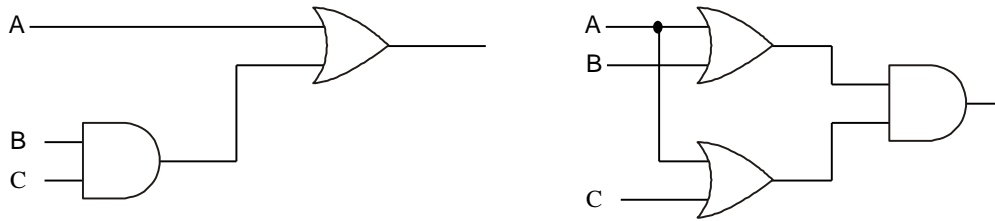
a)



b)



c)



DERIVING THE BOOLEAN EXPRESSION FROM A TRUTH TABLE

Consider the Truth Table given

A	B	Z
0	0	1
0	1	0
1	0	1
1	1	1

STEP 1

Note each combination that will give you a '1' at the output and write the Boolean expression for this line at the side of the Truth Table, next to the line that it applies to.

A	B	Z	
0	0	1	$\bar{A} \bullet \bar{B}$
0	1	0	
1	0	1	$A \bullet \bar{B}$
1	1	1	$A \bullet B$

STEP 2

Join the equations by putting an OR sign between each to give the final Boolean expression.

$$(\bar{A} \bullet \bar{B}) + (A \bullet \bar{B}) + (A \bullet B) = Z$$

ASSIGNMENT 6 b

1. Write the Boolean equation for the following truth tables

a)

A	B	Z
0	0	0
0	1	1
1	0	1
1	1	0

b)

A	B	C	Z
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

2. Develop a Boolean equation and draw a logic circuit diagram containing AND, OR and NOT gates to yield the truth table shown.

A	B	C	Z
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

DESIGNING A CIRCUIT FROM ITS BOOLEAN EQUATION

Consider the following equation

$$(\bar{A} \bullet \bar{B}) + (A \bullet \bar{B}) + (A \bullet B) = Z$$

STEP 1

Draw inputs A and B on the left-hand side of the page.

A

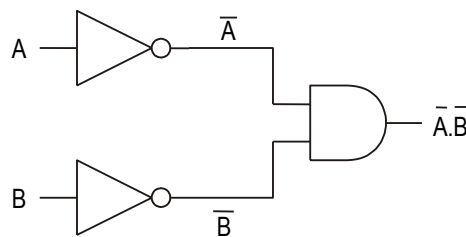
B

STEP 2

Start by considering the first term

$$\bar{A} \cdot \bar{B}$$

It can be seen from the equation that both inputs require to be inverted. This is achieved by passing signals from A and B through NOT gates. The outputs from these signals are then used as inputs to an AND.



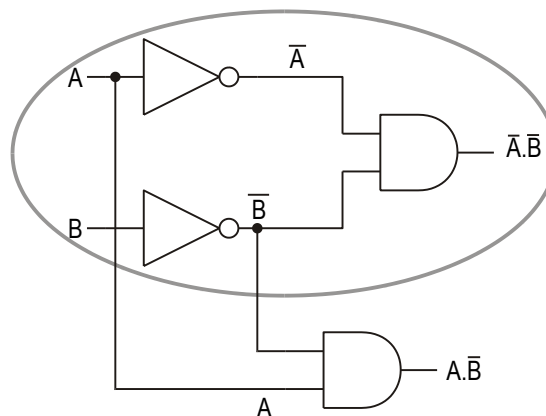
Write this on the output line of the AND gate.

STEP 3

Consider the next term

$$A \cdot \bar{B}$$

It can be seen from this term that input A does not require to be inverted but input B does. Since we already have both input A and B bar available we can amend the circuit diagram accordingly by feeding these inputs to a second AND gate.

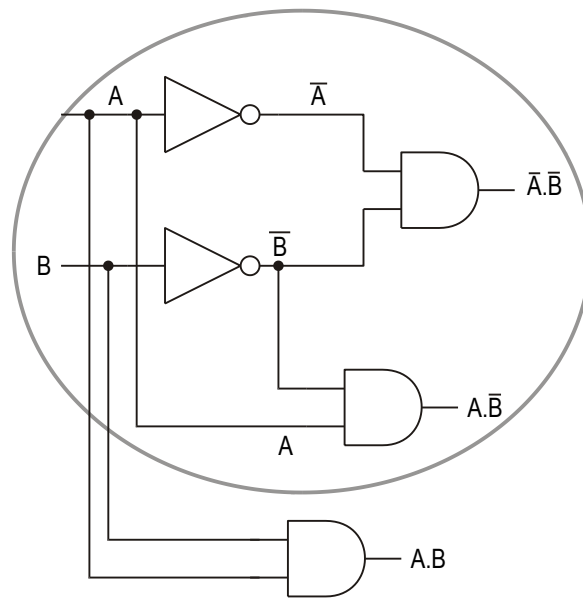


STEP 4

Consider the final term

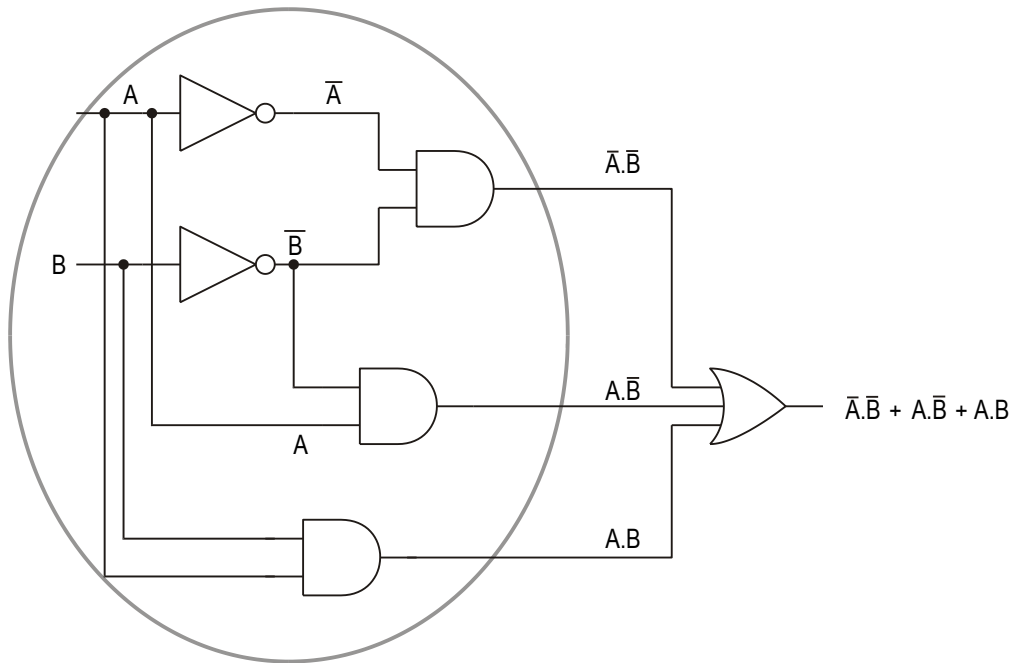
$$A \bullet B$$

Again both inputs are available to us. Neither of the inputs is inverted so the signals are taken from the original inputs and are fed to a third AND gate.



STEP 5

Finally we use the outputs from the three AND gates as inputs to a three input OR gate to arrive at the final solution.



ASSIGNMENT 7

1. For the following Boolean equations, draw the correct logic circuit arrangement.

- a) $A + (A \cdot B) = Z$
- b) $A + \bar{B} + C = Z$
- c) $(A \cdot B) + (A \cdot C) = Z$
- d) $(A \cdot B) + (\bar{A} \cdot C) + (B \cdot C) = Z$
- e) $(A \cdot B \cdot C) + (D \cdot E) + (F \cdot G) = Z$

COMBINATIONAL LOGIC CIRCUIT DESIGN

As problems become more and more difficult it is not always possible to go from the question to the answer in one or two steps, when that is the case the following set of rules should be followed.

When designing a system to suit a need you should proceed in the following order.

1. Describe the problem clearly in words.
2. Write out a Truth Table for the system.
3. Derive the Boolean expression from the Truth Table.
4. Simplify this expression if possible.
5. Draw a logic circuit diagram for the system using AND, OR and NOT gates.
6. Convert the circuit to NAND gates only.

It is entirely possible that not every problem will require all of these steps to be followed; however this will be a useful guide for most.

2. Write out a Truth Table for the system.

Inputs		green	red	alarm
A	B	light	light	bell
0	0	1	0	0
0	1	1	0	0
1	0	0	1	1
1	1	0	1	0

3. Derive the Boolean expression from the Truth Table.

Inputs		green		red		alarm	
A	B	light		light		bell	
0	0	1	$\bar{A} \bullet \bar{B}$	0		0	
0	1	1	$\bar{A} \bullet B$	0		0	
1	0	0		1	$A \bullet \bar{B}$	1	$A \bullet \bar{B}$
1	1	0		1	$A \bullet B$	0	

4. Simplify the expressions if possible.

$$\text{green light} - (\bar{A} \bullet \bar{B}) + (\bar{A} \bullet B) = Z$$

It can be seen that the output does not depend on the state of B since the output is HIGH when B is either HIGH or NOT HIGH i.e. the state of the green light depends only on the state of A, hence the expression can be simplified to

$$\text{green light} - \bar{A} = Z$$

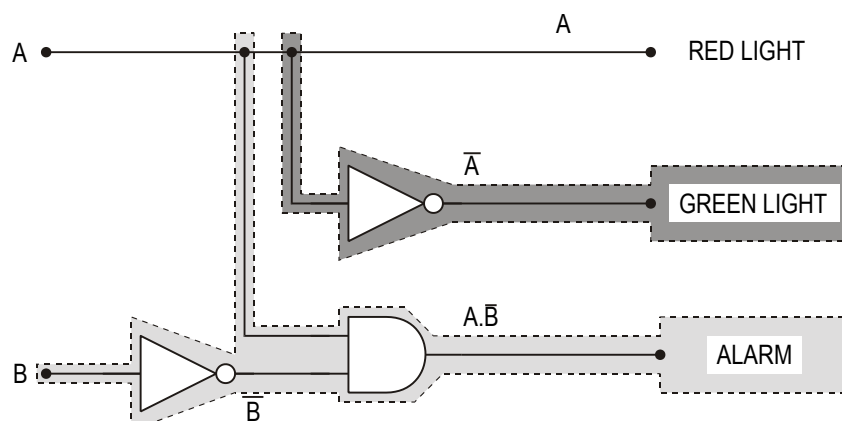
$$\text{red light} - (A \bullet \bar{B}) + (A \bullet B) = Z$$

Similarly it can be seen that the red light is ON irrespective of the state of B - the output only depends on the state of A hence this can be simplified to

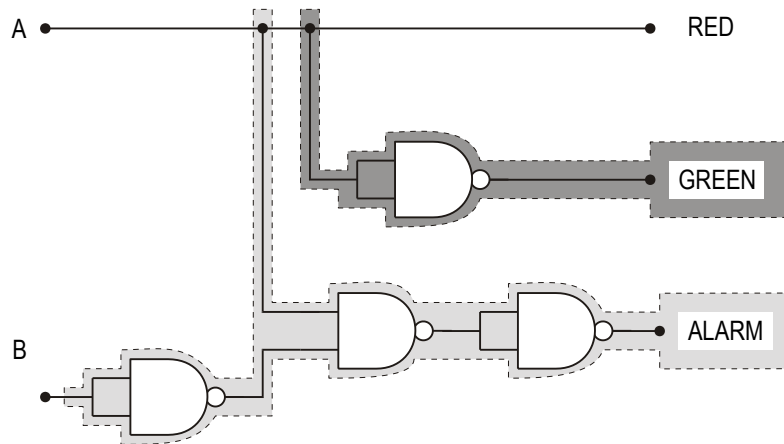
$$\text{red light} - A = Z$$

$$\text{alarm} - A \bullet \bar{B} = Z \text{ (can't be simplified further)}$$

5. Draw a logic circuit diagram for the system using AND, OR and NOT gates. Start with the expression that has fewest terms/gates (in this case $A=Z$, no gates!)



6. Convert the circuit to NAND gates only.



ASSIGNMENT 8

1. An electric guillotine must be adequately guarded. In order to safeguard the operator the machine has two switches, A and B, set about one metre apart, both of which need to be pressed before the machine will operate. Design a logic circuit that will give a green light if, both switches are not pressed and both switches are pressed. If only one switch is pressed a red light should come on. Assume that the switches return a 0 when not pressed and a 1 when pressed.

2. A given logic circuit has two logic inputs A and B. It is required to produce two logic outputs X and Y according to the following rules:

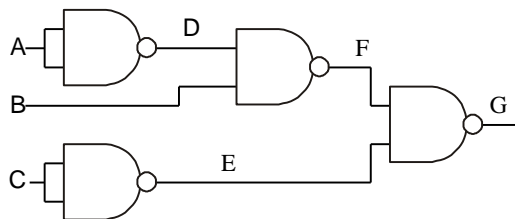
1. X is to be at logic 1 if (A OR B) but NOT (A AND B) are at logic 1.
2. Y is to be at logic 1 if (A AND B) but NOT (A OR B) are at logic 1.

Use the previous method to design an appropriate logic circuit.

3. A garage door is operated by a motor which is controlled by three switches.
The motor runs either:
when a pressure pad switch, A, in the drive is closed and a light dependent resistor switching circuit, B, is simultaneously activated by the car's headlights
or

when the key switch, C, in the garage door is operated.

- a) Prepare a truth table for all possible combinations of switching conditions for switched A, B and C. Take switch open as logic 0.
 - b) From the truth table, prepare a logic diagram using the least number of gates.
 - c) Redesign your circuit to use only 2-input NAND gates.
4. A domestic burglar alarm system is designed such that a bell will operate when the power switch is closed and a pressure switch under a carpet is closed or a switch is opened as a window is lifted.
- a) Assuming all switches to be logic state zero (0) when open
 - i) Draw a logic diagram for the design, allocating capital letters to the inputs to each gate and to the output to the bell.
 - ii) Prepare a truth table for the design. Your table must be headed by the appropriate letters
 - b) Show by use of a logic diagrams how you would modify or combine 3-input NAND gates to provide AND and OR gates.
5. The diagram below shows part of an industrial control system having three inputs A, B and C with an output G.



- a) How many different input conditions are possible in this system?
- b) What is the function of gates 1 and 2?
- c) Complete a truth table including columns which show the states of D, E, F and G.
- d) Why is the design made up entirely of NAND gates?
- e) Give an alternative design using 3-input AND and OR gates.