

## Topic: 1.3.1 Logic gates

Logic gates serve as the building blocks to digital logic circuits using combinational logic. Many electronic circuits operate using binary logic gates. Logic gates basically process signals which represent true or false or the equivalent i.e. ON or OFF, 1 or 0

Whilst there are a number of logic gates, only the six simplest are covered in this booklet:

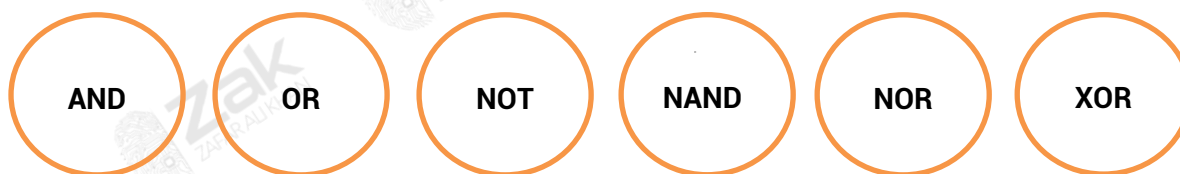
1. NOT gate
2. AND gate
3. OR gate
4. NAND gate
5. NOR gate
6. XOR gate.

The following notes describe the function of all six gates, how to produce truth tables, how to design networks using logic gates, and how to determine the output from a logic network

### The six main logic gates

The most common symbols used to represent logic gates are shown below. To avoid confusion the graphical representations will be used in exam questions but candidates may use either set of symbols when answering questions.







### Simple graphical representation



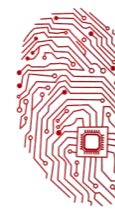


### Topic: 1.3.1 Logic gates

#### Symbols used to represent logic gates

Gate	Symbol	Operator
and		$A \cdot B$
or		$A + B$
not		$\bar{A}$
nand		$\overline{A \cdot B}$
nor		$\overline{A + B}$
xor		$A \oplus B$





### Topic: 1.3.1 Logic gates

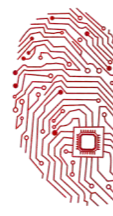
A **Truth Table** is simply a table listing all the combinations of inputs and their respective outputs.

The NOT gate has only one input, but the rest have 2 inputs.

The next section describes the function of all six logic gates.

Name	Symbol	Logic	Truth Table															
<b>NOT GATE</b>		The output (called X) is <b>true</b> (i.e. 1 or ON) when the <b>INPUT A</b> is <b>NOT TRUE</b> (i.e. 0 or OFF)	<table border="1"> <thead> <tr> <th>INPUT A</th> <th>OUTPUT X</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>0</td> </tr> <tr> <td>0</td> <td>1</td> </tr> </tbody> </table>	INPUT A	OUTPUT X	1	0	0	1									
INPUT A	OUTPUT X																	
1	0																	
0	1																	
<b>AND GATE</b>		The output is only <b>true</b> (i.e. 1 or ON) when the <b>(INPUT A AND INPUT B)</b> are both <b>TRUE</b> (i.e. 0 or OFF)	<table border="1"> <thead> <tr> <th>INPUT A</th> <th>INPUT B</th> <th>OUTPUT X</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> </tbody> </table>	INPUT A	INPUT B	OUTPUT X	1	1	1	1	0	0	0	1	0	0	0	0
INPUT A	INPUT B	OUTPUT X																
1	1	1																
1	0	0																
0	1	0																
0	0	0																
<b>OR GATE</b>		The output is <b>true</b> (i.e. 1 or ON) if <b>(INPUT A OR INPUT B)</b> are <b>TRUE</b> (i.e. 0 or OFF)	<table border="1"> <thead> <tr> <th>INPUT A</th> <th>INPUT B</th> <th>OUTPUT X</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> </tbody> </table>	INPUT A	INPUT B	OUTPUT X	1	1	1	1	0	1	0	1	1	0	0	0
INPUT A	INPUT B	OUTPUT X																
1	1	1																
1	0	1																
0	1	1																
0	0	0																
<b>NAND GATE</b>		<i>This is basically an AND gate with the output inverted</i> The output is <b>true</b> (i.e. 1 or ON) if <b>(INPUT A AND INPUT B)</b> are <b>NOT</b> both <b>TRUE</b> (i.e. 0 or OFF)	<table border="1"> <thead> <tr> <th>INPUT A</th> <th>INPUT B</th> <th>OUTPUT X</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> </tr> </tbody> </table>	INPUT A	INPUT B	OUTPUT X	1	1	0	1	0	1	0	1	1	0	0	1
INPUT A	INPUT B	OUTPUT X																
1	1	0																
1	0	1																
0	1	1																
0	0	1																
<b>NOR GATE</b>		<i>This is basically an OR gate with the output inverted</i> The output is <b>true</b> (i.e. 1 or ON) if <b>NOT (INPUT A AND INPUT B)</b> are <b>TRUE</b>	<table border="1"> <thead> <tr> <th>INPUT A</th> <th>INPUT B</th> <th>OUTPUT X</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> </tr> </tbody> </table>	INPUT A	INPUT B	OUTPUT X	1	1	0	1	0	0	0	1	0	0	0	1
INPUT A	INPUT B	OUTPUT X																
1	1	0																
1	0	0																
0	1	0																
0	0	1																





### Topic: 1.3.1 Logic gates

**EXCLUSIVE-OR GATE (XOR GATE)**



The output is true only when the inputs are opposite of each other

A	B	Output
0	0	0
0	1	1
1	0	1
1	1	0

The tables above containing 1s and 0s are known as truth tables and are an integral part of logic gates functionality. These are used extensively throughout this booklet in the design and testing of logic networks built up from logic gates.

#### Combinations of logic gates

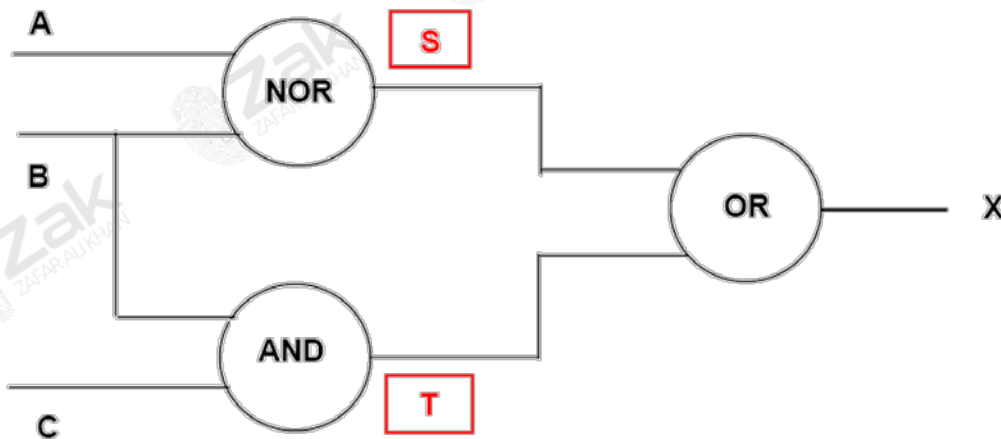
It is possible to combine logic gates together to produce more complex logic networks.

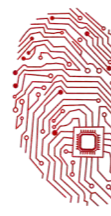
This booklet will only deal with a maximum of three inputs and up to six logic gates. The output from a logic network is checked by producing the truth table (as shown in the examples below).

We will deal with two different scenarios here. The first involves drawing the truth table from a given logic network and the second involves designing a logic network for a given problem and then testing it by drawing the truth table.

#### Producing the truth table from a given logic network

Consider the following logic network which contains three inputs and three logic gates:





### Topic: 1.3.1 Logic gates

If we now look at the output in two stages. First let us consider the outputs being produced at stages "S" and "T". To do this, we need to draw a truth table. There are three inputs (A, B and C) which gives 23 (i.e. 8) possible combinations of 1s and 0s. To work out the outputs at "S" and "T" we need to refer to the truth tables for the NOR gate and for the AND gate. For example, when A = 1 and B = 1 then we have 1 NOR 1 which gives the value of S = 0. Continuing doing the same thing for all 8 possible inputs we get the following **interim truth table**:

The final stage involves S OR T.

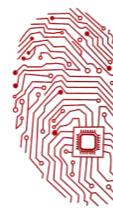
A	B	C	S	T
1	1	1	0	1
1	1	0	0	0
1	0	1	0	0
1	0	0	0	0
0	1	1	0	1
0	1	0	0	0
0	0	1	1	0
0	0	0	1	0

S	T	X
0	1	1
0	0	0
0	0	0
0	0	0
0	1	1
0	0	0
1	0	1
1	0	1

This gives the final truth table:

A	B	C	X
1	1	1	1
1	1	0	0
1	0	1	0
1	0	0	0
0	1	1	1
0	1	0	0
0	0	1	1
0	0	0	1





### Topic: 1.3.1 Logic gates

#### Designing logic networks to solve a specific problem and testing using truth tables

Consider the following problem:

**"If button A or button B are on and button C is off then the alarm X goes on"**

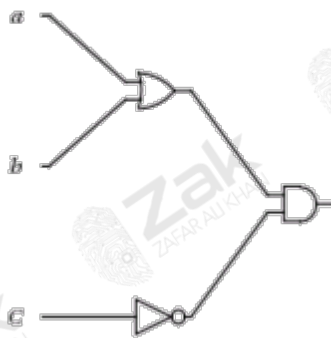
We can convert this into logic gate terminology (ON = 1 and OFF = 0):

**If (A = 1 OR B = 1) AND (C = NOT 1) then (X = 1)**

(Notice: rather than write 0 we use NOT 1)

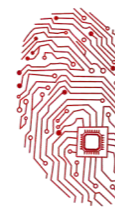
To draw the logic network, we do each part in brackets first i.e.  $A = 1 \text{ OR } B = 1$  is one gate then  $C = \text{NOT } 1$  is the second gate. These are then joined together by the AND gate. Once the logic network is drawn we can then test it using a truth table. Remember the original problem – we are looking for the output to be 1 when A or B is 1 and when C is 0. Thus we get the following logic network and truth table from the network. Looking at the values in the truth table, we will be able to clearly see that it matches up with the original problem which then gives us confidence that the logic network is correct.

Logic circuit:



A	B	C	X
1	1	1	0
1	1	0	1
1	0	1	0
1	0	0	1
0	1	1	0
0	1	0	1
0	0	1	0
0	0	0	0





### Topic: 1.3.1 Logic gates

Let us now consider a second problem:

A steel rolling mill is to be controlled by a logic network made up of AND, OR and NOT gates only. The mill receives a stop signal (i.e.  $S = 1$ ) depending on the following input bits:

INPUT	BINARY VALUE	CONDITION
L	1	Length > 100 metres
	0	Length < 100 metres
T	1	Temperature > 1000 C
	0	Temperature < 1000 C
V	1	Velocity > 10 m/s
	0	Velocity < 10 m/s

A stop signal ( $S = 1$ ) occurs when:

Either Length,  $L > 100$  meters and Velocity,  $V < 10$  m/s

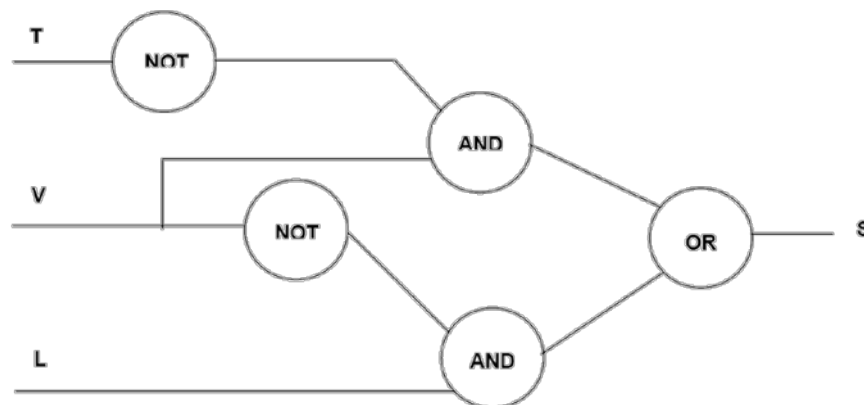
Or Temperature,  $T < 1000$  C and Velocity,  $V > 10$  m/s

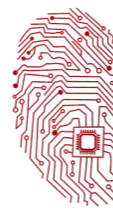
Draw a logic network and truth table to show all the possible situations when the stop signal could be received.

The first thing to do is to try and turn the question into a series of logic gates and then the problem becomes much simplified.

- Zak** The first statement can be re-written as: **( $L = 1$  AND  $V = \text{NOT } 1$ )** since Length > 100 meters corresponds to a binary value of 1 and Velocity < 10 m/s corresponds to a binary value of 0 (i.e. NOT 1).
- Zak** The second statement can be re-written as **( $T = \text{NOT } 1$  AND  $V = 1$ )** since Temperature < 1000C corresponds to a binary value of 0 (i.e. NOT 1) and Velocity > 10 m/s corresponds to a binary value of 1
- Zak** Both these statements are joined together by OR which gives us the logic statement:  
**if ( $L = 1$  AND  $V = \text{NOT } 1$ ) OR ( $T = \text{NOT } 1$  AND  $V = 1$ ) then  $S = 1$**

We can now draw the logic network and truth table to give the solution to the original problem (input L has been put at the bottom of the diagram just to avoid crossing over of lines; it merely makes it look neater and less complex and isn't essential):





### Topic: 1.3.1 Logic gates

L	T	V	S
1	1	1	0
1	1	0	1
1	0	1	1
1	0	0	1
0	1	1	0
0	1	0	0
0	0	1	1
0	0	0	0

