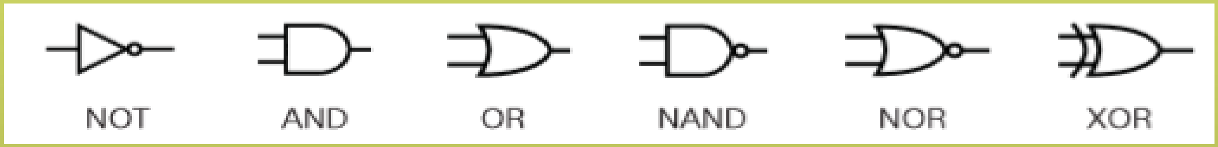
Syllabus Content:

1.3.1 Logic gates and logic circuits

- use logic gates to create electronic circuits
- understand and define the functions of NOT, AND, OR, NAND, NOR and XOR (EOR) gates, including the binary output produced from all the possible binary inputs (all gates, except the NOT gate, will have 2 inputs only)
- draw truth tables and recognise a logic gate from its truth table
- recognise and use the following standard symbols used to represent logic gates:

NOT AND OR NAND NOR XOR

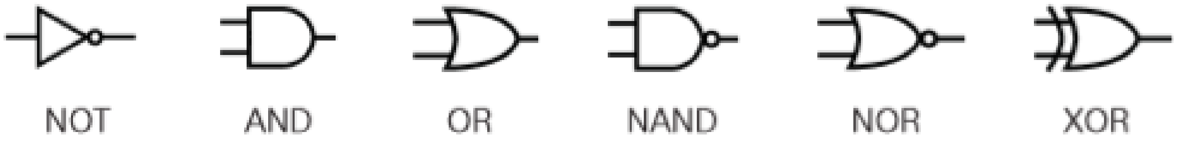


- produce truth tables for given logic circuits
- produce a logic circuit to solve a given problem or to implement a given written logic statement.

Logic gates and logic circuits

Introduction

Electronic circuits in computers, many new memories and controlling devices are made up of thousands of LOGIC GATES. Logic gates take binary inputs and produce a binary output.

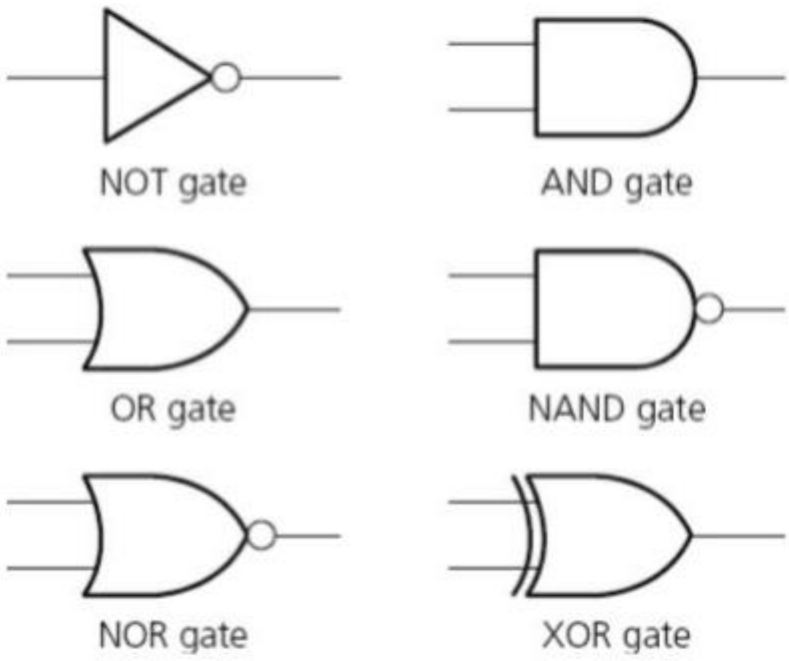


Several logic gates combined together form a LOGIC CIRCUIT and these circuits are designed to carry out a specific function. The checking of the output from a logic gate or logic circuit is done using a TRUTH TABLE.

This chapter will consider the function and role of logic gates, logic circuits and truth tables. Also a number of possible applications of logic circuits will be considered.

A reference to BOOLEAN ALGEBRA will be made throughout the chapter.

Six different logic gates will be considered in this chapter:



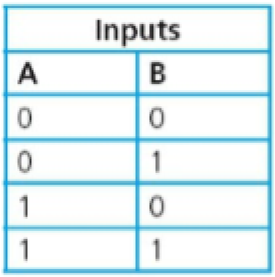
Truth tables:

Truth tables are used to trace the output from a logic gate or logic circuit.

The NOT gate is the only logic gate with one input; the other five gates have two inputs.

When constructing truth tables, all possible combinations of 1s and 0s which can be input are considered. For the NOT gate (one input) there are only **2^1** (2) possible binary combinations.

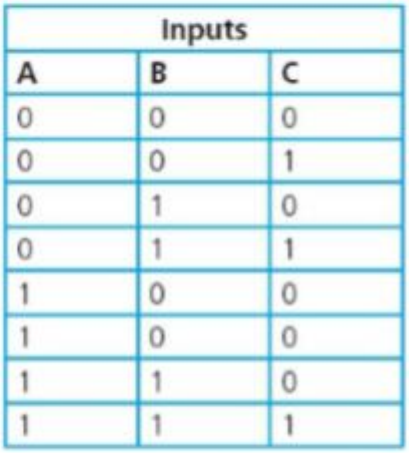
For all other gates (two inputs), there are **2^2 (4)** possible binary combinations.



Inputs	
A	B
0	0
0	1
1	0
1	1

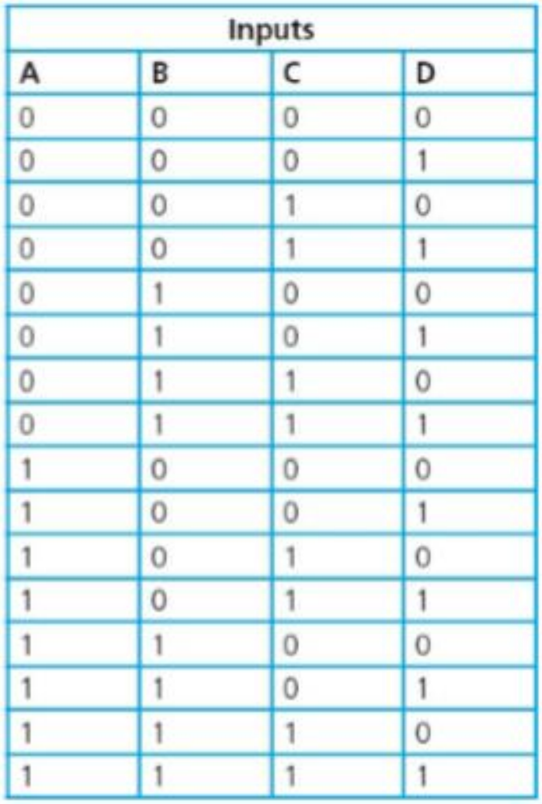
For logic circuits, the number of inputs can be more than 2;

For three inputs give a possible **2^3 (8)** binary combinations.

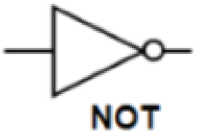


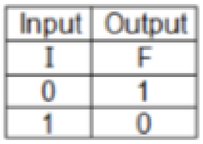
Inputs		
A	B	C
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	0
1	1	0
1	1	1

And for four inputs, the number of possible binary combinations is **2^4 (16)**. It is clear that the number of possible binary combinations is a multiple of the number 2 in every case.

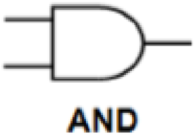


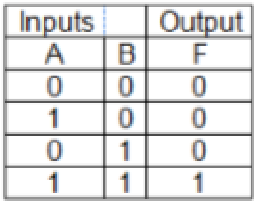
Inputs			
A	B	C	D
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1



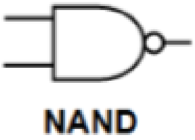


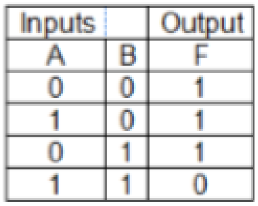
Input	Output
I	F
0	1
1	0



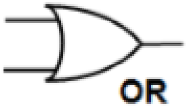


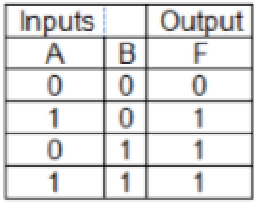
Inputs		Output
A	B	F
0	0	0
1	0	0
0	1	0
1	1	1



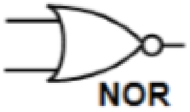


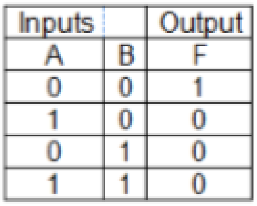
Inputs		Output
A	B	F
0	0	1
1	0	1
0	1	1
1	1	0



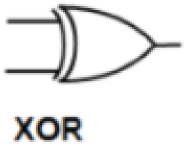


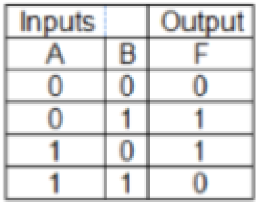
Inputs		Output
A	B	F
0	0	0
1	0	1
0	1	1
1	1	1





Inputs		Output
A	B	F
0	0	1
1	0	0
0	1	0
1	1	0





Inputs		Output
A	B	F
0	0	0
0	1	1
1	0	1
1	1	0

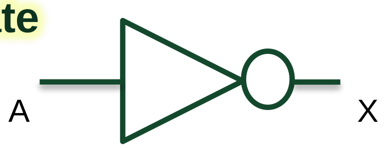
Logic Gates, Boolean Algebra and Truth Tables

Boolean Algebra is the mathematical foundation of digital circuits. Boolean Algebra specifies the relationship between Boolean variables which is used to design combinational logic circuits using Logic Gates. The truth table shows a logic circuit's output response to all of the input combinations.

- **Boolean Algebra**
  - A Boolean Variable takes the value of either 0 (False) or 1 (True).
  - Symbols are used to represent Boolean variables e.g. **A, B, C, X, Y, Z**
  - There are three basic logic operations **AND**, **OR**, **NOT**
  - The Boolean Operators are • + ¯
    - **A + B means A OR B**
    - **A • B means A AND B**
    - **$\overline{A}$ means NOT A**
  - Nodes in a circuit are represented by Boolean Variables

The function of the logic gates

NOT gate

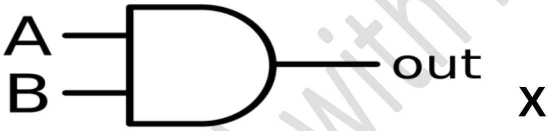


Input	Output
A	**X**
0	1
1	0

X = NOT A

X = $\overline{A}$

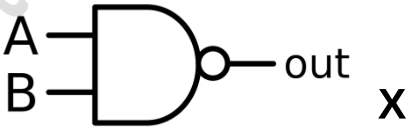
AND gate



Input A	Input B	Output X
0	0	0
0	1	0
1	0	0
1	1	1

X = A AND B (logic notation)
X = $A \cdot B$ (Boolean algebra)

NAND gate

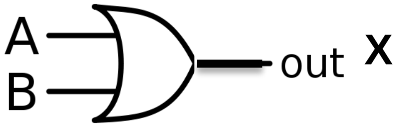


Input A	Input B	Output X
0	0	1
0	1	1
1	0	1
1	1	0

X = A AND B (logic notation)

X = $\overline{A \cdot B}$ (Boolean algebra)

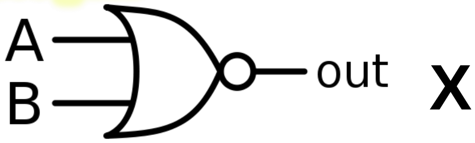
OR gate



Input A	Input B	Output X
0	0	0
0	1	1
1	0	1
1	1	1

X = A OR B (logic notation)
X = A + B (Boolean algebra)

NOR gate

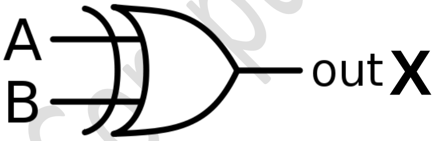


Input A	Input B	Output X
0	0	1
0	1	0
1	0	0
1	1	0

X = A NOR B (logic notation)

X = $\overline{A + B}$ (Boolean algebra)

XOR gate



Input A	Input B	Output X
0	0	1
0	1	0
1	0	0
1	1	1

X = A XOR B (logic notation)

X = $A . \overline{B} + \overline{A} . B$ (Boolean algebra)

Logic circuits:

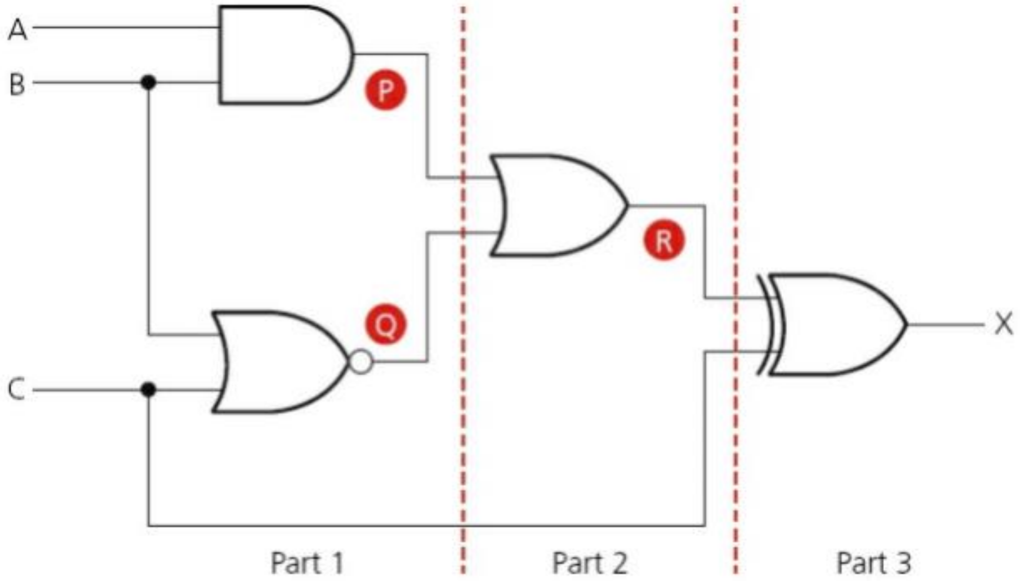
When logic gates are combined together to carry out a particular function, such as controlling a robot, they form a logic circuit. The output from the logic circuit is checked using a truth table.

There now follows three examples which show:

- how to produce a truth table
- how to design a logic circuit from a given logic statement/Boolean algebra
- how to design a logic circuit to carry out an actual safety function.

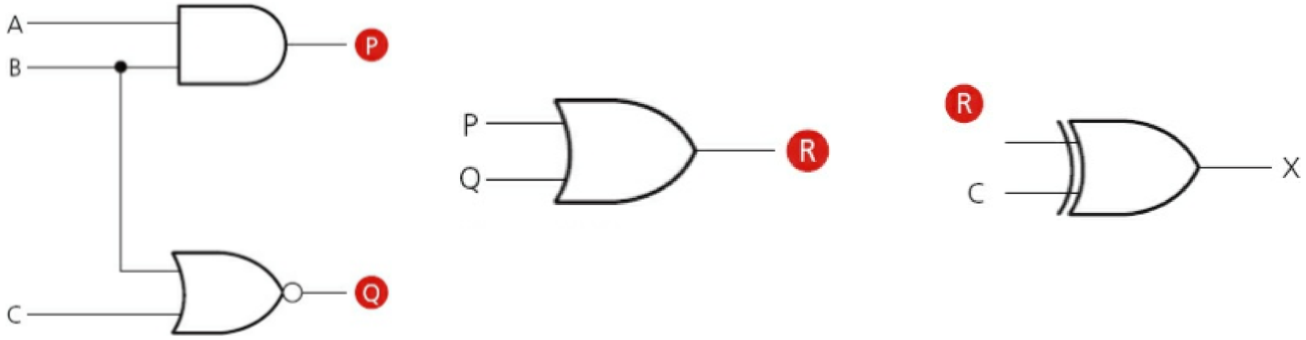
Example 1

Produce a truth table for the following logic circuit (note the use of • at junctions):

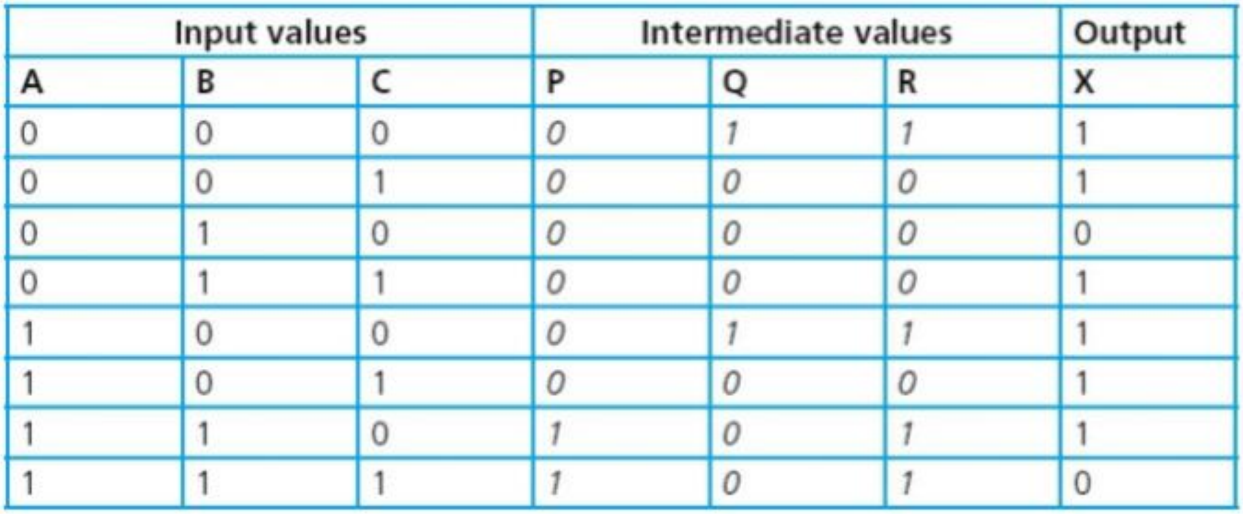


There are three inputs to this logic circuit, therefore there will be eight possible binary values which can be input.

To show step-wise how the truth table is produced, the logic circuit has been split up into three parts and intermediate values are shown as P, Q and R.



The truth table for the logic circuit will look like this:

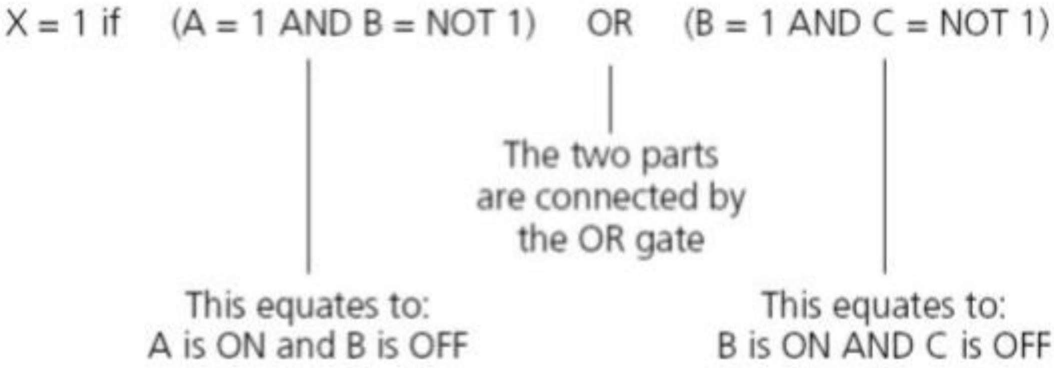


Input values			Intermediate values			Output
A	B	C	P	Q	R	X
0	0	0	*0*	*1*	*1*	1
0	0	1	*0*	*0*	*0*	1
0	1	0	*0*	*0*	*0*	0
0	1	1	*0*	*0*	*0*	1
1	0	0	*0*	*1*	*1*	1
1	0	1	*0*	*0*	*0*	1
1	1	0	*1*	*0*	*1*	1
1	1	1	*1*	*0*	*1*	0

Example 2:
A safety system uses three inputs to a logic circuit. An alarm, X, sounds if input A represents ON and input B represents OFF; or if input B represents ON and input C represents OFF.

Produce a logic circuit and truth table to show the conditions which cause the output X to be 1.

The first thing to do is to write down the logic statement representing the scenario in this example. To do this, it is necessary to recall that ON = 1 and OFF = 0 and also that 0 is usually considered to be NOT 1. So we get the following logic statement:



X = 1 if (A = 1 AND B = NOT 1) OR (B = 1 AND C = NOT 1)

The two parts are connected by the OR gate

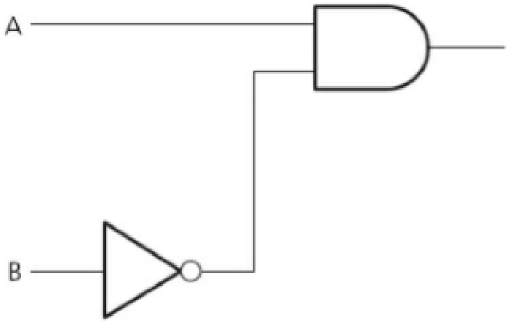
This equates to: A is ON and B is OFF

This equates to: B is ON AND C is OFF

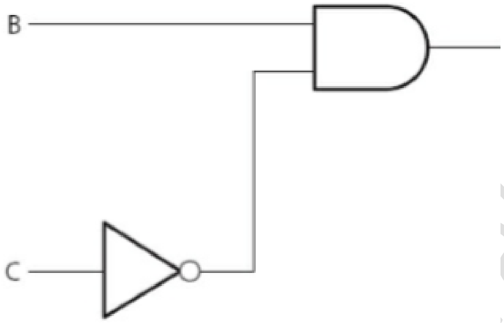
Note: this statement can also be written in Boolean algebra as:

$$(a \cdot \overline{b}) + (b \cdot \overline{c})$$

The logic circuit is made up of three parts as shown in the logic statement.
We will produce the logic gate for the first part and the third part. Then join both parts together with the OR gate.

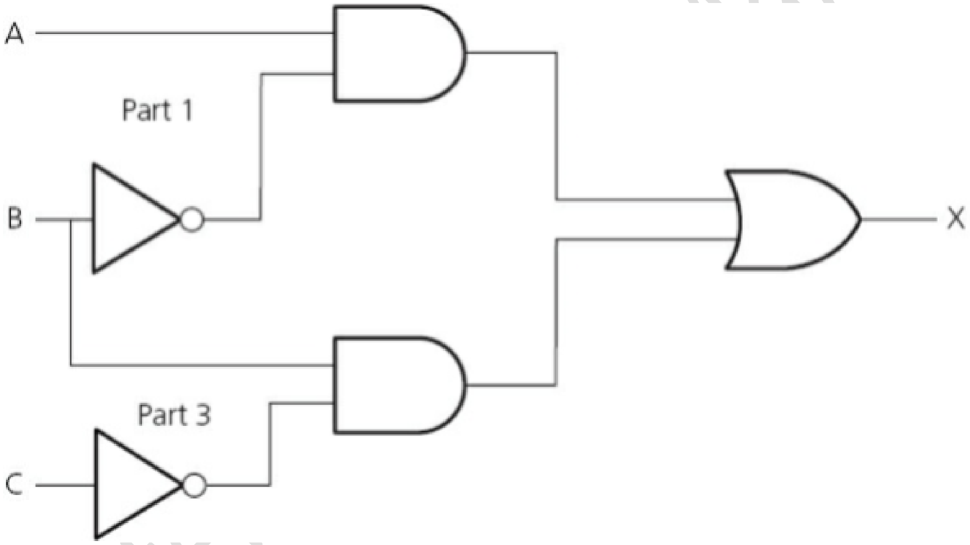


Part 1



Part 3

Now combining both parts with the OR gate gives us:

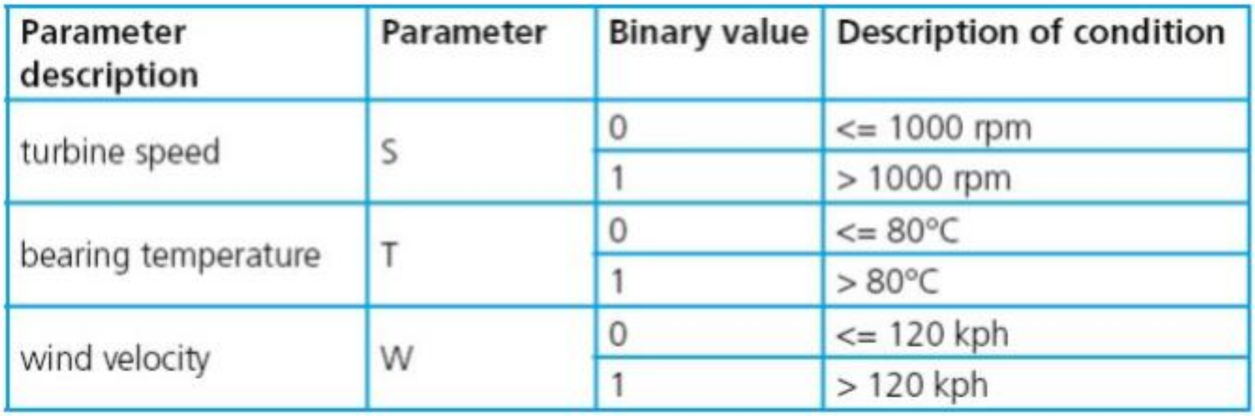


Inputs			Intermediate values		Output
A	B	C	(A=1 AND B=NOT 1)	(B=1 AND C=NOT 1)	X
0	0	0	*0*	*0*	0
0	0	1	*0*	*0*	0
0	1	0	*0*	*1*	1
0	1	1	*0*	*0*	0
1	0	0	*1*	*0*	1
1	0	1	*1*	*0*	1
1	1	0	*0*	*1*	1
1	1	1	*0*	*0*	0

Example 3:

A wind turbine has a safety system which uses three inputs to a logic circuit. A certain combination of conditions results in an output, X, from the logic circuit being equal to 1. When the value of X = 1 then the wind turbine is shut down.

The following table shows which parameters are being monitored and form the three inputs to the logic circuit.



Parameter description	Parameter	Binary value	Description of condition
turbine speed	S	0	<= 1000 rpm
		1	> 1000 rpm
bearing temperature	T	0	<= 80°C
		1	> 80°C
wind velocity	W	0	<= 120 kph
		1	> 120 kph

The output, X, will have a value of 1 if any of the following combination of conditions occur:

- either turbine speed <= 1000 rpm and bearing temperature > 80°C
- or turbine speed > 1000 rpm and wind velocity > 120 kph
- or bearing temperature <= 80°C and wind velocity > 120 kph.

Design the logic circuit and complete the truth table to produce a value of X =1 when any of the three conditions above occur.

This is a different type of problem to those covered in Examples 1 and 2. This time a real situation is given and it is necessary to convert the information into a logic statement and then produce the logic circuit and truth table.

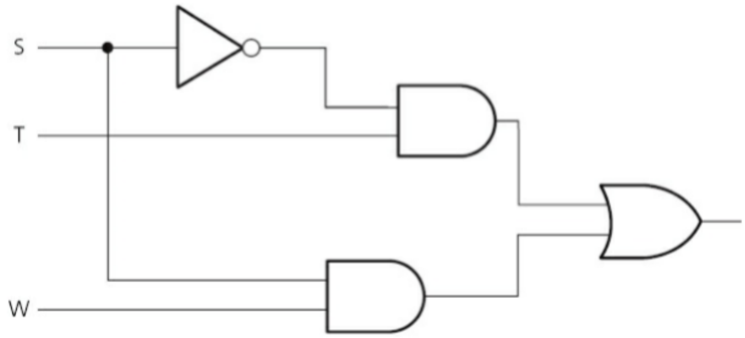
Stage 1:

The first thing to do is to convert each of the three statements into logic statements. Use the information given in the table and the three condition statements to find how the three parameters, S, T and W, are linked.

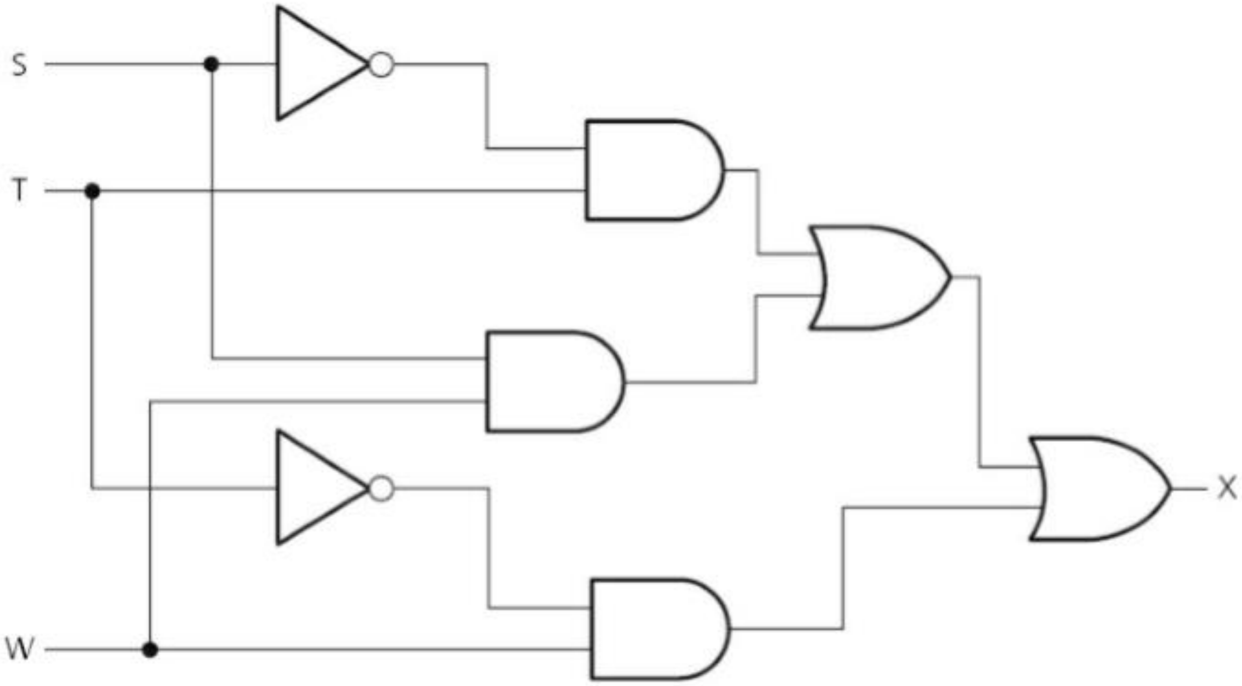
We usually look for the key words **AND, OR** and **NOT** when converting actual statements into logic. We end up with the following three logic statements:

1. turbine speed <= 1000 rpm and bearing temperature > 80°C
   logic statement: **(S = NOT 1 AND T = 1)**
2. turbine speed > 1000 rpm and wind velocity > 120 kph
   logic statement: **(S = 1 AND W = 1)**
3. bearing temperature <= 80°C and wind velocity > 120 kph
   logic statement: **(T = NOT 1 AND W = 1)**

We will start by joining (**1**) and (**2**) together using an OR gate:



Finally, we connect the logic circuit in **1,2 to 3** to obtain the answer:

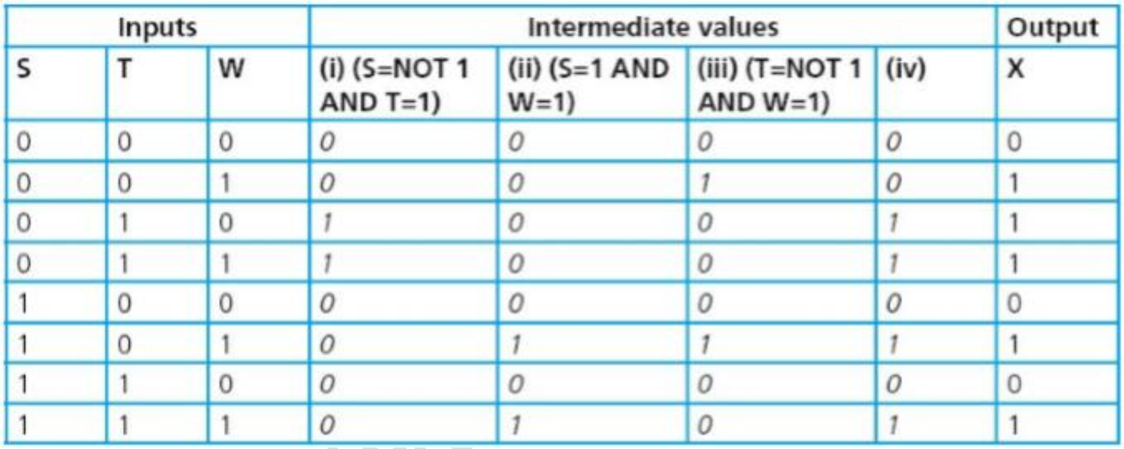


The final part is to produce the truth table. We will do this using the original logic statement. This method has the bonus of allowing an extra check to be made on the logic circuit to see whether or not it is correct.
It is possible, however, to produce the truth table straight from the logic circuit. There were three parts to the problem, so the truth table will first evaluate each part.

Then, by applying OR gates, as shown below, the final value, X, is obtained:

- **i.** (S = NOT 1 AND T = 1)
- **ii.** (S = 1 AND W = 1)
- **iii.** (T = NOT 1 AND W = 1)

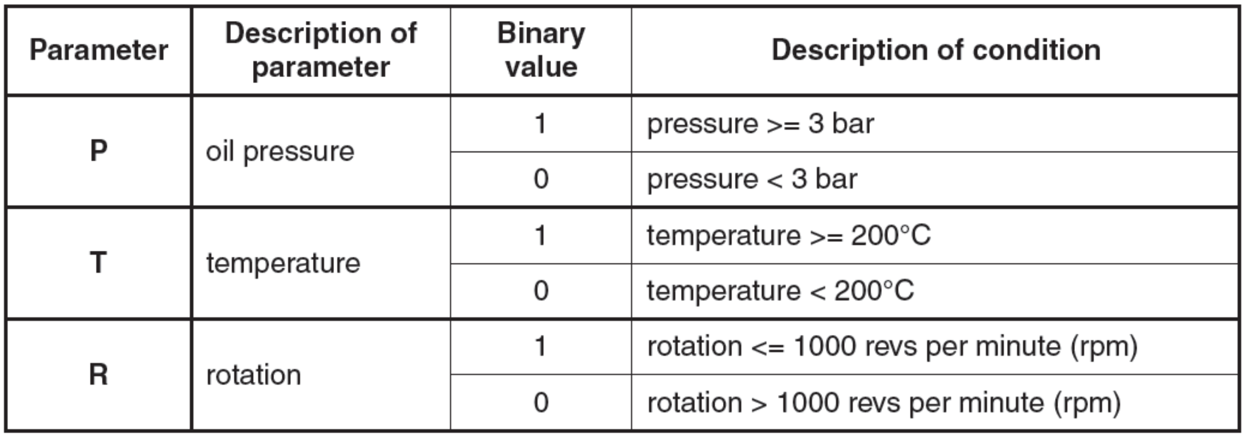
We find the outputs from parts (i) and (ii) and then OR these two outputs together to obtain a new intermediate, which we will label part (iv). We then OR parts (iii) and (iv) together to get the value of X.



Inputs			Intermediate values				Output
S	T	W	(i) (S=NOT 1 AND T=1)	(ii) (S=1 AND W=1)	(iii) (T=NOT 1 AND W=1)	(iv)	X
0	0	0	*0*	*0*	*0*	*0*	0
0	0	1	*0*	*0*	*1*	*0*	1
0	1	0	*1*	*0*	*0*	*1*	1
0	1	1	*1*	*0*	*0*	*1*	1
1	0	0	*0*	*0*	*0*	*0*	0
1	0	1	*0*	*1*	*1*	*1*	1
1	1	0	*0*	*0*	*0*	*0*	0
1	1	1	*0*	*1*	*0*	*1*	1

Exam questions:

Q.1 A system is monitored using sensors. The sensors output binary values corresponding to physical conditions, as shown in the table:

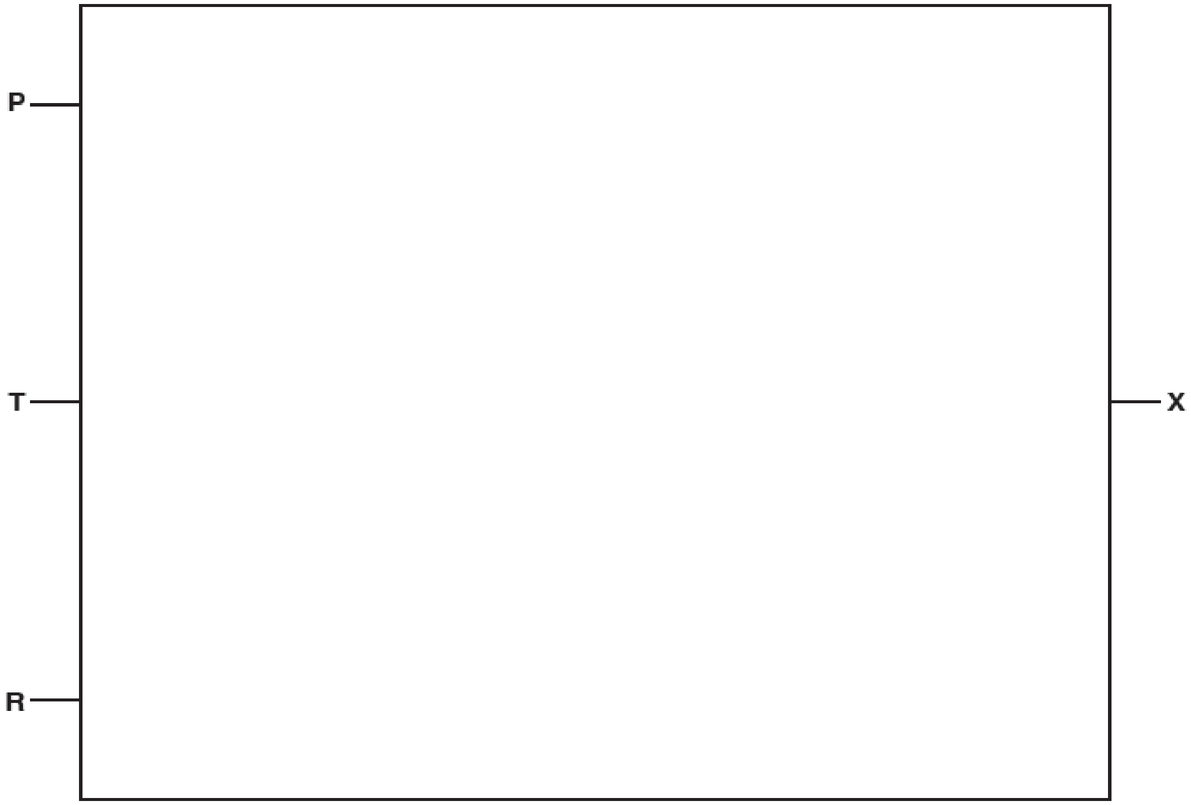


Parameter	Description of parameter	Binary value	Description of condition
P	oil pressure	1	pressure >= 3 bar
		0	pressure < 3 bar
T	temperature	1	temperature >= 200°C
		0	temperature < 200°C
R	rotation	1	rotation <= 1000 revs per minute (rpm)
		0	rotation > 1000 revs per minute (rpm)

The outputs of the sensors form the inputs to a logic circuit. The output from the circuit, X, is 1 if any of the following three conditions occur:

either oil pressure >= 3 bar **and** temperature >= 200°C
or oil pressure < 3 bar **and** rotation > 1000 rpm
or temperature >= 200°C **and** rotation > 1000 rpm

(a) Draw a logic circuit to represent the above system.



[5]

(b) Complete the truth table for this system.

P	T	R	Workspace	X
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

[4]

EXAM STYLE QUESTIONS

Q.1 (a) A student needs to design a logic circuit to model the requirements for membership of a snooker club. Membership (X) depends on four criteria, as shown in the table:

Parameter	Description of parameter	Binary value	Condition
A	Over 18	1	True
		0	False
B	Recommended	1	True
		0	False
C	Full-time	1	True
		0	False
D	Retired	1	True
		0	False

Membership is approved (X = 1) if the person:

- is over the age of 18 and has been recommended by a pre-existing member and
- either is working full-time or is retired, but not both.

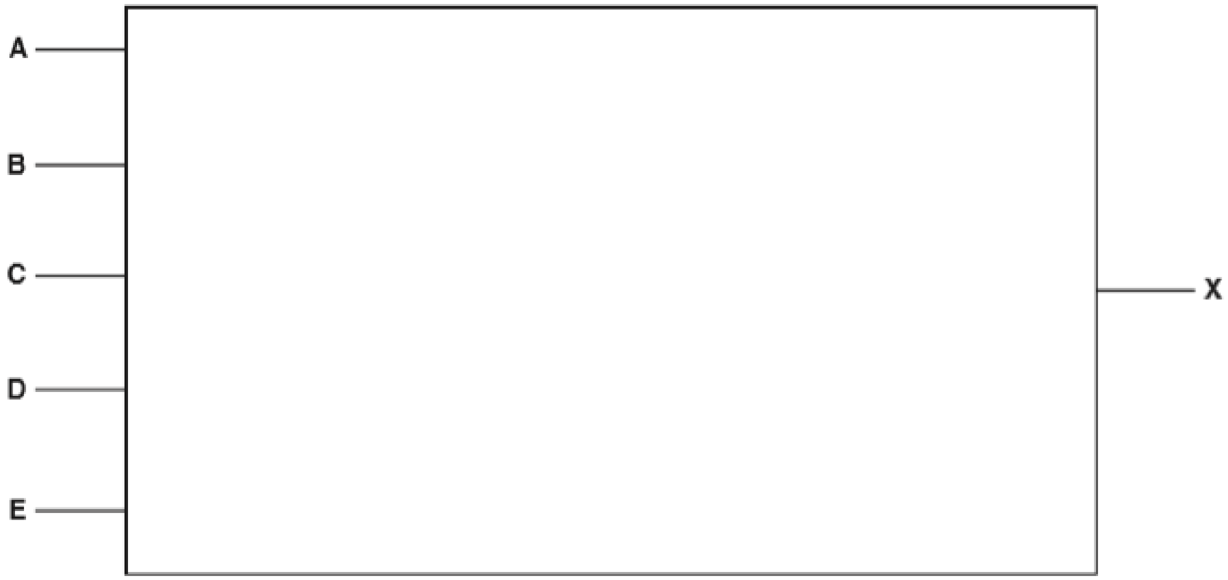
Draw a logic circuit to represent the membership requirements.

[3]

Q2 (a) An alarm system (X) is enabled and disabled using either a switch (A) or a remote control (B). There are two infra-red sensors (C, D) and one door pressure sensor (E).

Parameter	Description of parameter	Binary value	Condition
A	Switch	1	Switch enabled
		0	Switch disabled
B	Remote control	1	Remote enabled
		0	Remote disabled
C	Infra-red sensor	1	Activated
		0	Not activated
D	Infra-red sensor	1	Activated
		0	Not activated
E	Door pressure sensor	1	Activated
		0	Not activated

The alarm sounds (X = 1) if the alarm is enabled and any one or more of the sensors is activated. Draw a logic circuit to represent the alarm system.



[3]

Electronics companies need to consider the cost of components, ease of fabrication and time constraints when designing and building logic circuits.

We will mention two possible ways electronics companies can review logic circuit design:

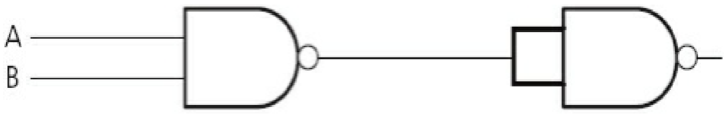
- One method is to use 'off-the-shelf' logic units and build up the logic circuit as a number of 'building blocks'.
- Another method involves simplifying the logic circuit as far as possible; this may be necessary where room is at a premium (e.g. in building circuit boards for use in satellites to allow space exploration).

Using logic 'building blocks'

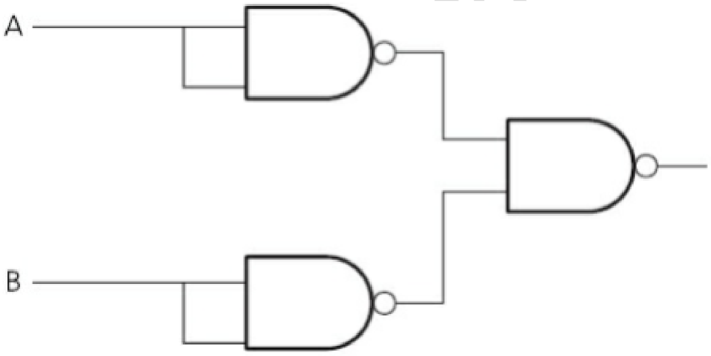
One very common 'building block' is the **NAND** gate. It is possible to build up any logic gate, and therefore any logic circuit, by simply linking together a number of **NAND** gates.

For example, the AND, OR, NOT and XOR gates can be built from these gates as shown below:

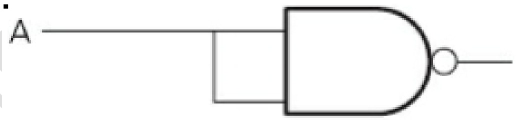
The AND gate:



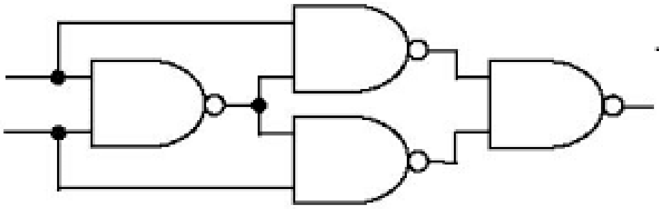
The OR gate:



The NOT gate:



The XOR gate: Exclusive OR (XOR)

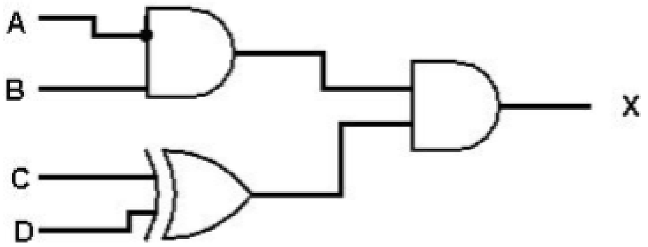


A	B	Output
0	0	0
0	1	1
1	0	1
1	1	0

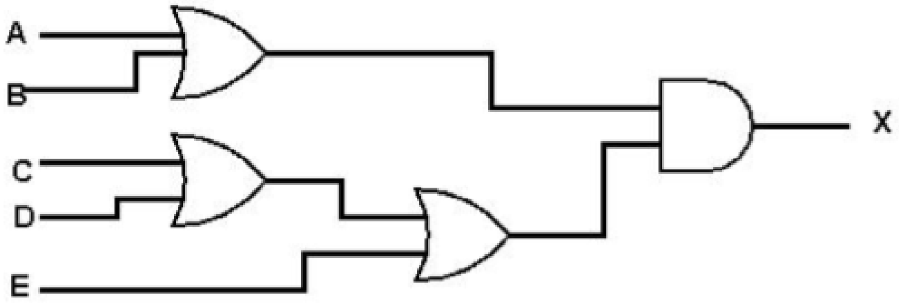
Answer:

EXAM STYLE QUESTIONS

Q.1.

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>5(a)</td><td><b>1 mark</b> per correct gate with correct inputs
 </td><td>3</td></tr>
<tr><td>5(b)</td><td><b>1 mark</b> for each correct pair of lines
<table>
<tr><th>A</th><th>B</th><th>C</th><th>Working space</th><th>X</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td></td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td></td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td></td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td></td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td></td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td></td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td></td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td></td><td>0</td></tr>
</table></td><td>4</td></tr>
</table>

Q2

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>4(a)</td><td>1 mark per bullet:
□ A OR B
□ C OR D OR E
□ Final AND
 </td><td>3</td></tr>
<tr><td>4(b)</td><td><b>1 mark</b> for each correct pair of rows
<table>
<tr><th>A</th><th>B</th><th>C</th><th>Working space</th><th>X</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td></td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td></td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td></td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td></td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td></td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td></td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td></td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td></td><td>1</td></tr>
</table></td><td>4</td></tr>
</table>

References:
IGCSE Computer Science by Hodder Education.
Past papers.
http://electronics-course.com/logic-gates