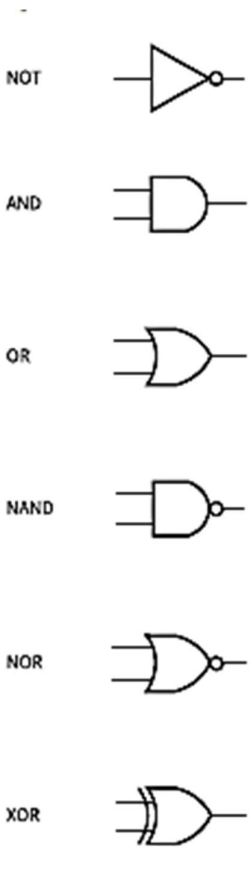
Logic gates

What is a logic gate?

A logic gate is a **visual way** of representing a **Boolean expression**



A	X
0	1
1	0

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

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

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

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

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

Figure 4.02 Logic gate symbols and their associated truth tables

There are two other points to note here. The NOT gate is a special case, having only one input. The second point is that a NAND gate is a combination of a AND gate followed by a NOT gate, and a NOR gate is a combination of an OR gate followed by a NOT gate. NAND and NOR gates produce a complementary output to the AND and OR gates.

Logic construction From problem statements

- A problem statement is typically a **written interpretation of a scenario** that requires a **specific logical outcome**
- Logic circuits can be **constructed using from problem statements**

Worked Example

A server room has an **automatic cooling fan (F)**
The fan turns on based on the following input parameters:

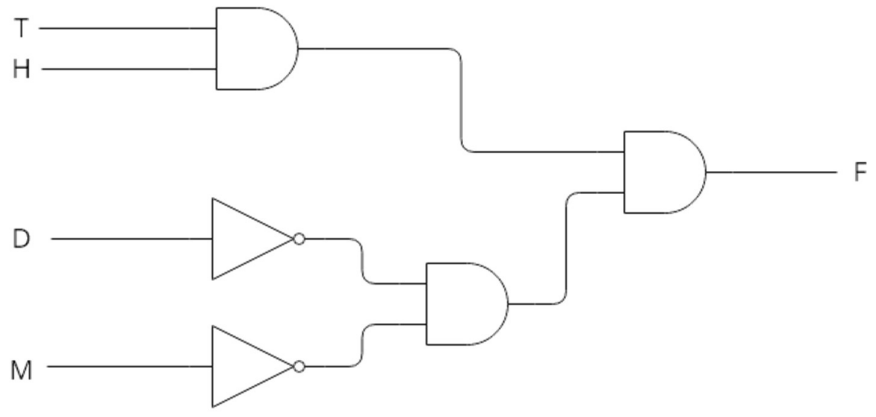
Parameter	Description	Binary Value	Condition
T	Temperature	1 = Too high	0 = Acceptable
H	Humidity	1 = Too high	0 = Acceptable
M	Maintenance mode	1 = Active	0 = Inactive
D	Door sensor	1 = Door open	0 = Door closed

The **fan (F = 1)** turns on if:

- The **temperature is too high**
- **AND** humidity is too high
- **AND** the **door is closed**
- **UNLESS** maintenance mode is active (if maintenance is active, the fan must stay off)

Draw a logic circuit to represent how the fan (F) operates based on the input conditions. [3]

Answer



NOT gates on **D** and **M** inputs [1 mark]
AND gate combines **T AND H** or **NOT D AND NOT M** [1 mark]
Combines two previous AND outputs (e.g. (**T AND H) AND (NOT D AND NOT M)**) [1 mark]

From logic expressions

- A logic expression is a way of showing how a **logic circuit works using symbols** and **Boolean logic** (AND, OR, NOT)
- It describes the conditions that must be met for the **output** to be **true (1)** or **false (0)**
- From a logic expression, a **logic circuit and/or truth table can be constructed**

Worked Example

A logic expression is given:

S = (A AND B AND C) OR (B XOR C)

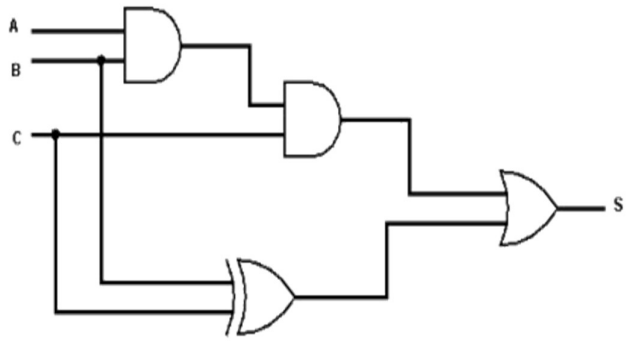
(a) Draw the logic circuit for the given expression **[4]**

(b) Complete the truth table for the logic expression: **[2]**

S = (A AND B AND C) OR (B XOR C)

Answer

(a)



- Each correct gate [1 mark]

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

(b)
Rows **1 to 4** correct [1 mark]
Rows **5 to 8** correct [1 mark]

From truth tables

- To create a truth table for the expression **P = (A AND B) AND NOT C**
  - Calculate the numbers of rows needed ($2^{\text{number of inputs}}$)
  - In this example there are 3 inputs (**A, B, C**) so a total of **8 rows** are needed (2^3)
  - To not miss any combination of inputs, start with **000** and **count** up in **3−bit binary (0−7)**

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

- Add a new column to show the **results** of the brackets first (A AND B)
- Add a new column to show the **results** of NOT C
- The last column shows the **result** of the Boolean expression (**P**) by comparing (A AND B) AND NOT C

A	B	C	A AND B	NOT C	P
0	0	0	0	1	0
0	0	1	0	0	0
0	1	0	0	1	0
0	1	1	0	0	0
1	0	0	0	1	0
1	0	1	0	0	1
1	1	0	1	1	0
1	1	1	1	0	0

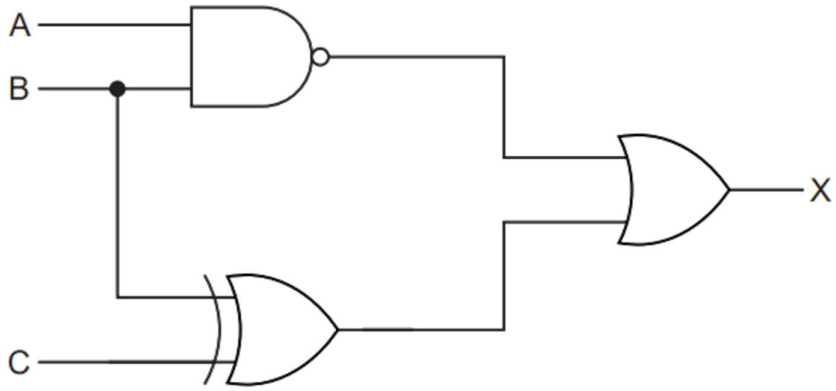
Examiner Tips and Tricks
It is possible to create a truth table when combining expressions that show only the inputs and the final outputs.
The inclusion of the extra columns supports the process but can be skipped if you feel able to do those in your head as you go.

From logic circuits

- From a logic circuit you can create the **logic expression and/or truth table**

Worked Example

Write the logic expression for the given logic circuit. **[3]**



Answer

(A NAND B) OR (B XOR C)

- A NAND B [1 mark]
- B XOR C [1 mark]
- OR [1 mark]