AQA Computer Science A-Level

4.6.4 Logic gates

Advanced Notes

<u>Specification:</u>

4.6.4.1 Logic gates:

Construct truth tables for the following logic gates:

- NOT
- AND
- OR
- XOR
- NAND
- NOR

Be familiar with drawing and interpreting logic gate circuit diagrams involving one or more of the above gates.

Complete a truth table for a given logic gate circuit.

Write a Boolean expression for a given logic gate circuit.

Draw an equivalent logic gate circuit for a given Boolean expression.

Recognise and trace the logic of the circuits of a half-adder and a full-adder.

Construct the circuit for a half-adder.

Be familiar with the use of the edge-triggered D-type flip-flop as a memory unit.

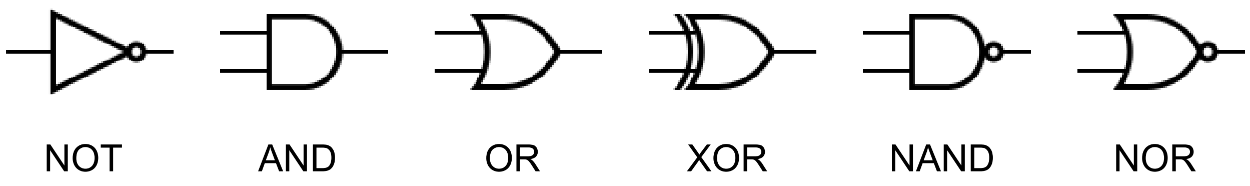
Logic Gates

A computer's processor is made up of billions of logic gates, devices which apply logical operations to one or more Boolean inputs in order to produce a single output.

Within a processor, logic gates are combined to form logic circuits. These can perform more complex operations like binary addition.

Logic Gate Symbols

Each of the six required logic gates has an internationally recognised symbol which you should learn. The symbols have inputs on the left and outputs on the right.



Truth Tables

A truth table shows every possible combination of inputs and the corresponding output for a logic gate or logic circuit. The inputs are labelled alphabetically starting with A and the output is usually labelled Q.

NOT

The NOT gate has one input and one output. The gate's output is always the opposite of its input. If the input to the gate is a 1, it will output 0 and vice versa.

A	Q
0	1
1	0

The truth table for the NOT gate has just two columns, the input A and the output Q. There are just two possible inputs, 1 and 0.

$$Q = \overline{A}$$

Note

In Boolean logic, an **overline** signifies that the value(s) below have the **NOT** operation applied to them.

AND

The AND gate has two inputs, labelled A and B in the truth table below, and outputs the product of the two inputs.

A	B	Q
0	0	0
0	1	0
1	0	0
1	1	1

$$Q = A \times B$$

The AND gate only outputs TRUE (1) when both inputs are TRUE, otherwise it outputs FALSE.

OR

In the same way that AND multiplies its inputs, OR adds them together. Therefore, the OR gate is said to output the sum of its inputs.

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	1

$$Q = A + B$$

OR only outputs FALSE when both inputs are FALSE. When one or more of the gate's inputs are TRUE, the logic gate outputs TRUE.

XOR

The XOR gate's full name is exclusively or and it outputs TRUE when strictly one of its inputs is TRUE. The gate's truth table is the same as the OR gate with the exception of the last line in which FALSE is output with two TRUE inputs.

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0

Synoptic Link

The **XOR** operation is used in the **Vernam cipher**.

The Vernam cipher is covered in more detail in **representing images, sound and other data** under **fundamentals of data representation**.

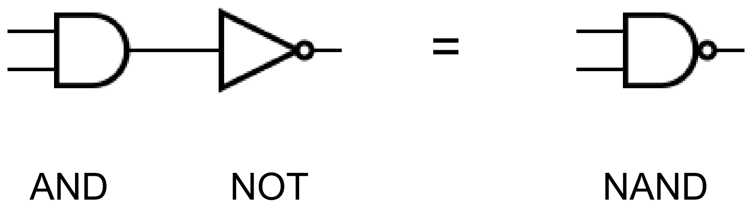
$$Q = A \oplus B$$

Note

XOR has its own symbol, which is a combination of an **O** and a **+** sign.

NAND

NAND is short for NOT AND. The NAND gate is actually a combination of two gates which we've already covered, the NOT gate and the AND gate.



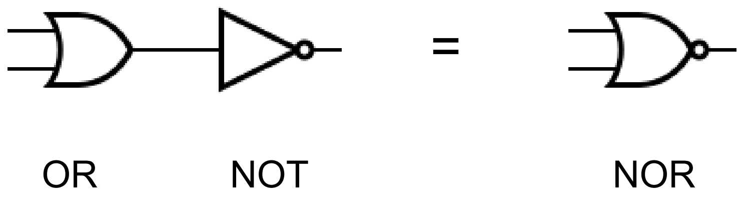
The NAND gate's truth table is the same as the AND gate's truth table, but the output is reversed.

A	B	Q
0	0	1
0	1	1
1	0	1
1	1	0

$$Q = \overline{A \times B}$$

NOR

NOR, short for NOT OR is a combination of the two logic gates NOT and OR.



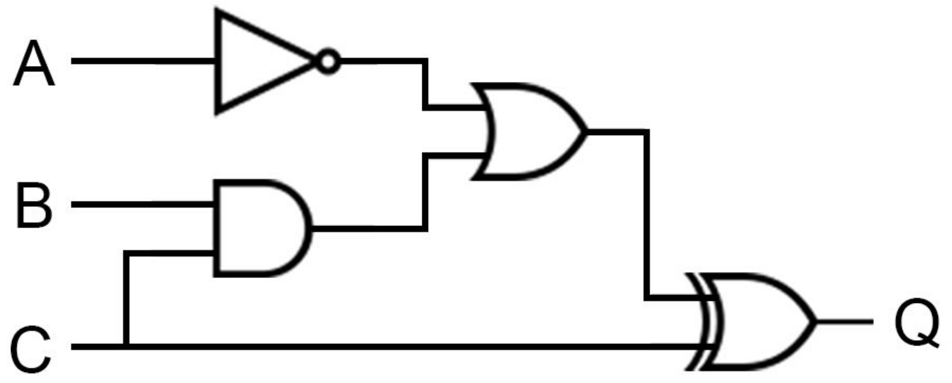
Therefore, the NOR gate's truth table is the same as the OR gate's table, just with the output reversed.

A	B	Q
0	0	1
0	1	0
1	0	0
1	1	0

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

Combining Logic Gates

Logic gates can be combined to form more complex circuits. You may be asked to draw or interpret a logic circuit involving multiple logic gates.



The logic circuit above combines four logic gates and can be represented using the logical expression below.

Synoptic Link

There are more examples of **logical expressions** in the notes for **Boolean algebra**.

$$Q = C \oplus ((B \times C) + \overline{A})$$

In order to create a truth table for this circuit, we first need to fill in all the possible permutations of inputs like so:

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

Notice that the input column furthest to the left alternates between 0 and 1, the next column to the right alternates between 00 and 11 and the column furthest to the right has four 0s followed by four 1s. You'll see this convention used time and time again with truth tables.

Next, we add columns for each of the elements that make up the logical expression such as $B \times C$ and $\overline{A}$. This will make it easier for us to combine them to form the final expression.

A	B	C	$B \times C$	$\overline{A}$	$(B \times C) + \overline{A}$	$C \oplus ((B \times C) + \overline{A})$
0	0	0	0	1	1	1
0	0	1	0	1	1	0
0	1	0	0	1	1	1
0	1	1	1	1	1	0
1	0	0	0	0	0	0
1	0	1	0	0	0	1
1	1	0	0	0	0	0
1	1	1	1	0	1	0

Once the column in the truth table for the finished expression is complete, the columns used for working can be removed and the final column renamed Q.

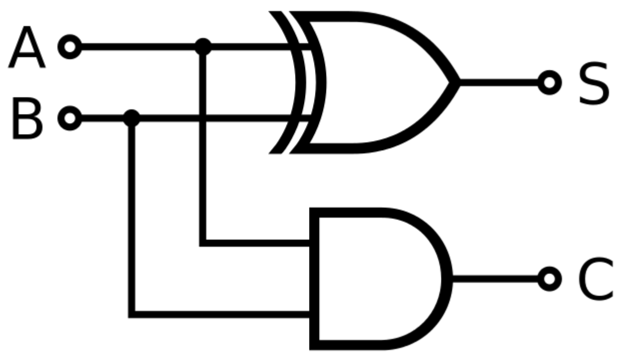
A	B	C	Q
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

Adders

An adder is a logic circuit that can be used to add Boolean values together. There are two types of adder that you need to be aware of: half adders and full adders.

Half adders

A half adder is a logic circuit two inputs, two outputs and two logic gates. The circuit can be used to add two Boolean values.



The two inputs are labelled A and B and the outputs are labelled S and C. Short for sum and carry.

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

0 + 0 = 0

0 + 1 = 1

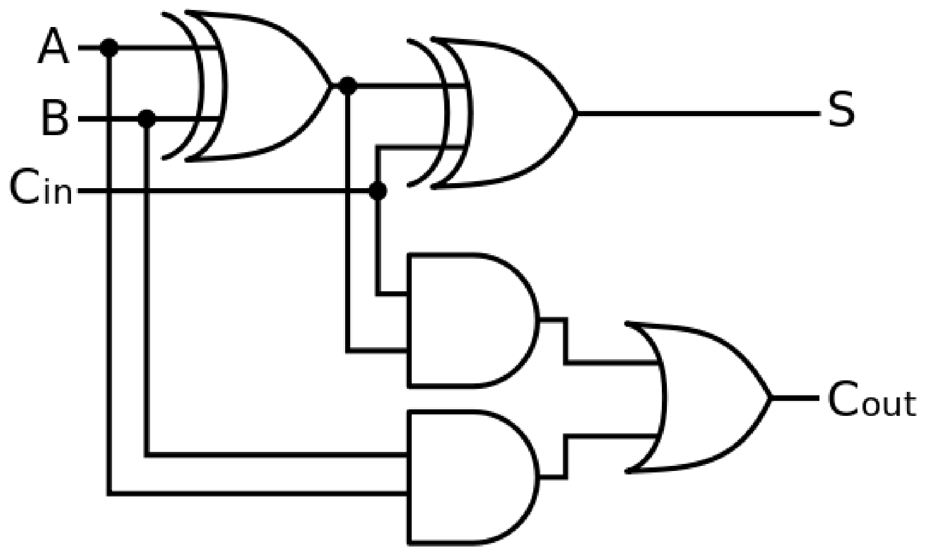
1 + 0 = 1

1 + 1 = 1 carry 1

You need to be able to draw the logic circuit for a half adder.

<u>Full adders</u>

A full adder has three inputs and two outputs, enabling it to input two Boolean values and a carry bit from a previous, less significant operation.



The three inputs are labelled A, B and C_{in} for carry in. The two outputs are labelled S for sum and C_{out} for carry out.

You need to be able to recognise this circuit as a full adder, but you're not expected to be able to draw it.

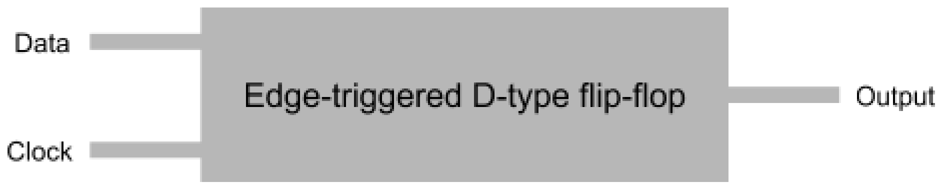
The full adder's truth table looks like this:

A	B	C_{in}	S	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Edge-triggered D-type flip-flop

An edge-triggered D-type flip-flop is a logic circuit which can be used as a memory unit for storing the value of a single bit.

An edge-triggered D-type flip-flop has two inputs, one for data and another for a clock signal. There is one output, which always holds the value of the stored bit.



The clock signal is generated by the computer and alternates between 0 and 1 at a set frequency. The value of the stored bit is set to the value of the data input with each change of the clock signal.

The clock signal can also be used to synchronise numerous flip-flops when they form part of a larger system such as a shift register.