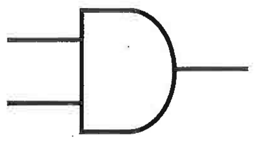
Division of Computing Science and Mathematics, University of Stirling

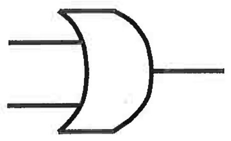
CSCU9V4 Systems
Tutorial 1a: Basic Logic Gates and Truth Tables

In this tutorial we will understand the functioning of the logic gates and circuits that form the basis of most of the internal workings of a CPU, cache memory and other devices in a computer. These logic gates have standardised names and symbols as follows:

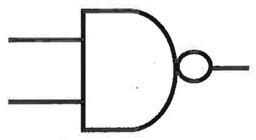
AND gate



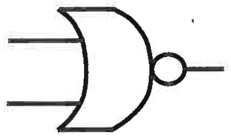
OR gate



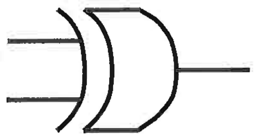
NAND gate



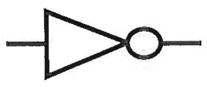
NOR gate



XOR gate



NOT gate



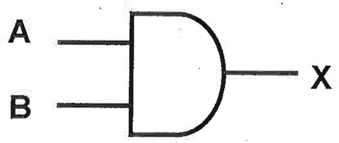
You need to memorise these, and the truth tables from lecture slides! Keep this tutorial sheet handy in your notes until you know them completely. To understand better you can build the circuits on the simulators (links provided earlier) and check the output.

Remember that computing devices use <u>Boolean logic</u>, which consists of two states - TRUE and FALSE. This is implemented in real circuits by using 5v and 0v respectively to represent those states. For simplicity we shall use 1 and 0 to represent these states in the truth tables and in the Boolean algebra.

Tasks (Use your chosen simulator)

1. Using a wire, connect a switch to a light and check that both logic states (on and off) can be produced.
2. By connecting switches to the inputs (let's call them A, B) and an LED to the output (X) of each gate, produce **truth tables** for the following logic gates. The truth table for the AND gate has been included to get you started - notice that two inputs means there are 4 possible combinations of inputs to that gate (i.e. 2^2), and these are shown as the 4 rows in the table. You need to fill in the X column:

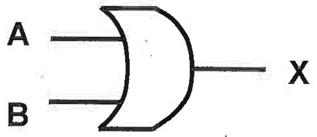
AND



Truth table for AND gate:

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

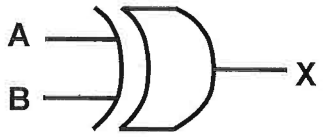
OR



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

XOR

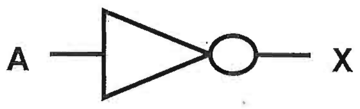
(exclusive OR)



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

NOT

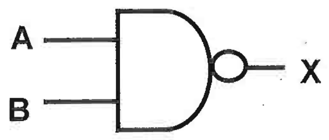
(inverter)



A	X
0	1
1	0

NAND

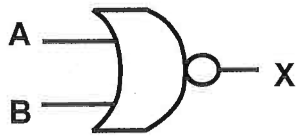
(Not AND)



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

NOR

(Not OR)

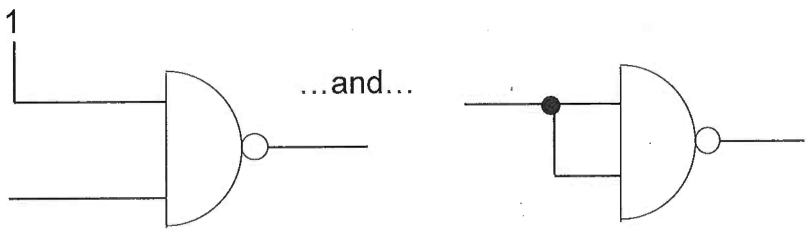


A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

3. Many of the basic logic functions seen earlier can be built using only **NAND** (Not AND) gates. This means that if you can create a NAND gate you can go on to build a computer.

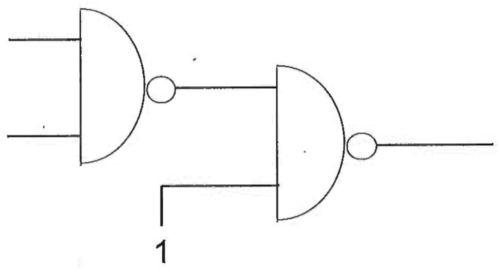
   Build the circuits below and draw up a truth table for each, noting the LED outputs for each set of inputs (as for task 2), and so determine which basic logic gate each circuit is equivalent to (e.g. by comparing the truth table against those derived in task 2).

(A)



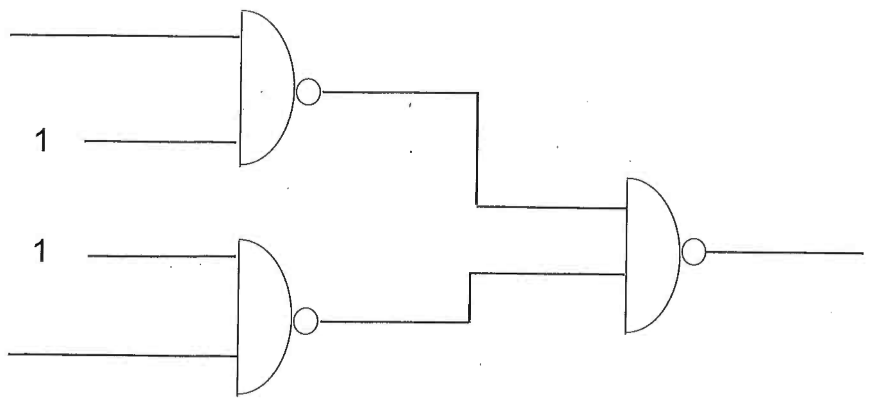
Both equivalent to a: NOT gate

(B)



Equivalent to a: AND gate

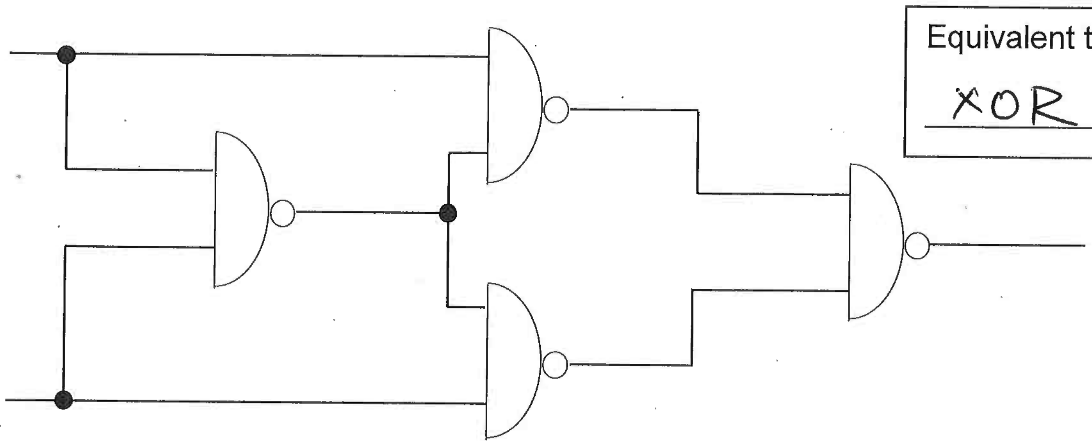
(C)



Equivalent to a:

OR gate

(D)



Equivalent to a:

XOR gate