# Chapter 19: Logic circuits and Boolean algebra: Answers to coursebook

## Worksheet 19.1: for testing basic understanding

**1** **a** A combinational circuit is one where the output depends only on the inputs. A sequential circuit is one where the output depends both on the inputs and on the previous output.

**b** Both types of circuit use the standard logic gates. However, for sequential circuits there is often a component, sometimes referred to as a state device, which acts as a memory unit storing the previous output. In addition, some sequential circuits are synchronous: they contain a clock to regulate activity by means of a clock pulse.

**c** Adders are the most obvious examples of combinational circuits; others are multiplexers and encoders. Flip-flops are the obvious examples of sequential circuits.

**2** **a**

| A | B | X |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

This could be worked out by inserting intermediate points after each of the first three logic gates. However, it is possible to see the result by realising that when the two inputs to a NOR gate come from the same source, the gate acts as an inverter (a NOT gate). The single NOR gate in the middle of the circuit produces a 1 only if both inputs are 0, which happens when the original inputs are both 1.

**b** A NAND gate.

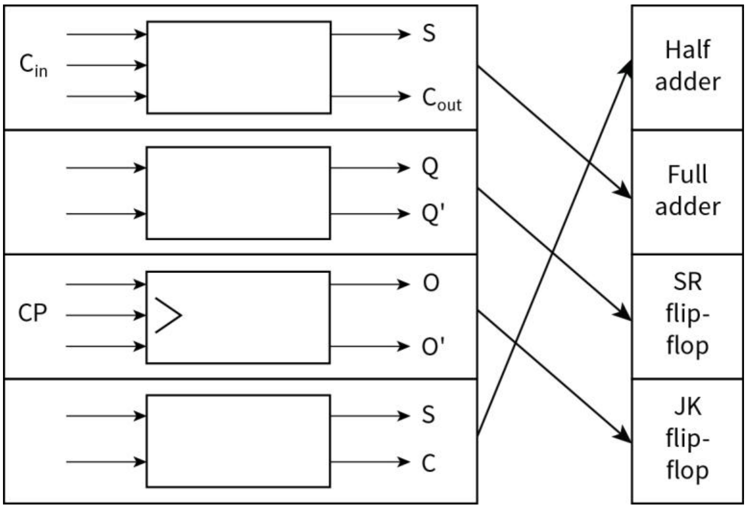
**c** NOR and NAND gates are universal gates. Any circuit can be created by using only NAND gates or only NOR gates.

**d** If NAND gates were already available there would be no need to construct this circuit!

**3** A True, B False, C False, D True, E True

The extra output from an adder circuit is the value of the carry resulting from the addition. Only the full adder circuit, which has three inputs, can use the carry from a previous adder output. Note that any circuit can be constructed from just NAND gates or from just NOR gates. Both of these are examples of universal gates.

**4**




**5** **a** C or D, A or B, F or G, E or H

This is an RS flip-flop created using NOR gates, so only an on signal can cause a change. The truth table usually presented does not provide the full story. The flip-flop is only in a useful state when storing the value 1 or 0 on the Q output with the Q’ output having the inverse value. This state stays stable if R and S are reading 0.

**b** C or D, because both inputs are 0 and the initial state is acceptable.

**c** A or B, because the initial state has two 0 values.

**d** F or G, because one input is 1 but initial and final states are the same.

For E the input is S = 1 and R = 0 so the final state will be Q = 1 and Q’ = 0, but this will not be immediately reached. Examination of the circuit shown shows that Q = 1 is only output if R = 0 and Q’ = 0 and that Q’ is only output if S = 0 and Q = 0. This means that an intermediate state with Q = 0 and Q’ = 0 is reached when a signal is applied but this is not a stable state so the signal causes a follow-on change.

**6**

- $\overline{A}.B + A.\overline{B} + A.B$ 1
- $A + B$ 3
- $\overline{A}.B + A.\overline{B} + A.B + A.B$ 2
- $A.1 + 1.B$ 4
- $A.(B + \overline{B}) + (A + \overline{A}).B$ 5

The method here illustrates a common approach. It takes an expression and duplicates one of the terms to produce pairs of terms that can then be simplified. The law written as A + A = A can be reversed so that A can be changed to A + A with the understanding that A can be absolutely anything.

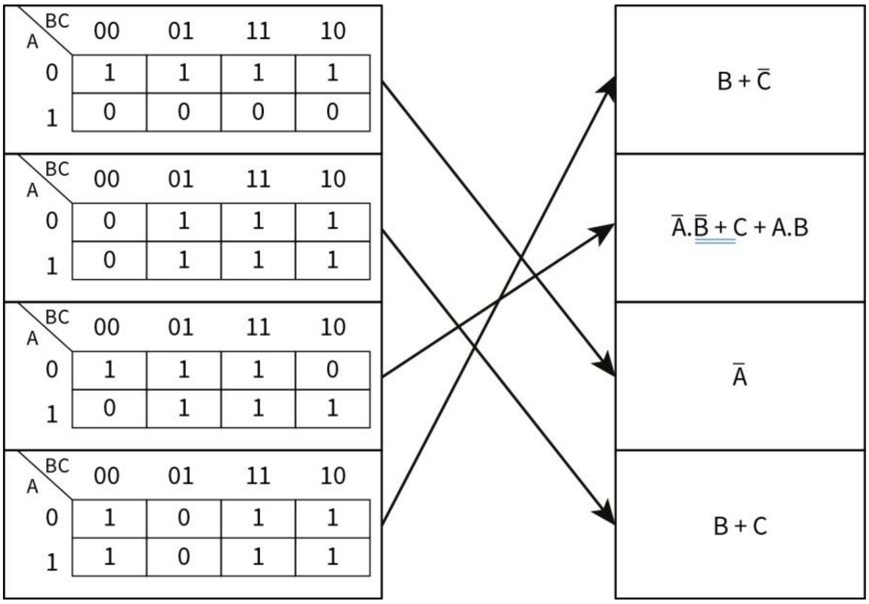
**7** **a**

| AB \ BC | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 |

**b** There are two overlapping pairs of cells that each contain two 1 values. These give the combination:

A.B + A.C = A.(B + C)

**8**




It is necessary to apply the rules here carefully. Every block of 2, 4, 8 filled with 1s has to be identified. This means that some cells fit into more than one group. Note that there is wrap around when either of the rows or columns represents two values as it does here for BC heading the columns. Because the columns hold two values there can be wrap around from the end of a row to the beginning of the same row. It is essential to use the full number of cells each time. Two cells must not be treated as one group if they are attached to two other cells.

Having chosen a group, there has to be a check on which value or values such as A or $\overline{C}$ do not change throughout the group. Values that do not change are ANDed together, which means A.B for example. If the number of groups is more than one, the Boolean algebra expressions are ORed together, for example A.B + B.

# Worksheet 19.2: more challenging questions

**1** **a** For the top left gate, the logic is A + A = A then applying the NOT gives $\overline{A}$.

For the bottom left gate, similar logic gives $\overline{B}$.

The output from the single NOR gate is $\overline{A} + \overline{B}$ followed by the NOT operation, giving:

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

Applying De Morgan gives AB (or A.B in dot notation).

The final gate then produces $\overline{AB}$.

**b** The truth table shows the logical output as A NAND B which is A AND B followed by NOT. Boolean algebra writes A AND B as AB.

**c** NOT, as mentioned in the answer to question 2a in Worksheet 19.1

**2** **a**

| AB \ CD | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 0 | 0 | 0 | 0 |
| 01 | 0 | 1 | 1 | 1 |
| 11 | 1 | 0 | 0 | 1 |
| 10 | 0 | 0 | 0 | 0 |

**b** From the AB = 11 row using wrap-around we get $AB\overline{D}$

From the two overlapping pairs in the AB = 01 row we get $\overline{A}BD$ and $\overline{A}BC$.

From the CD = 10 column we get $BC\overline{D}$

So the expression for X is:

$AB\overline{D} + \overline{A}BD + \overline{A}BC + BC\overline{D}$