Overview

In this unit your students should:

- learn how to design logic systems from just NAND gates
- practice assembling logic systems out of tested sub-systems
- meet the use of a seven-segment LED dsiplay
- learn how to analyse NAND gate systems with Boolean algebra
- appreciate the advantages of building logic systems from just one type of gate

This should not require more than 4 hours of class time.

Hour	Suggested Activity
1	Launch them straight into the **NOR from NAND** practical. As they finish, get them to answer the questions on page 2 of the **Only NAND gates** exercises. Ask them to complete steps 1 to 4 of the **Multiplexer** practical before the next session.
2	Let them go straight on to the **Multiplexer** practical. As they finish, they could go on to the **Binary to seven-segment converter** practical. They are unlikely to get beyond step 3 in the time available. Ask them to complete the design work of steps 1 to 3 before the next session.
3	Students should spend this session assembling and testing the sub-systems of the **Binary to seven-segment converter** practical. Some students may get as far as the design work for step 5. They could answer the questions on pages 64 and 65 of the text book before the next session, leaving it free for them to assemble and test their design.
4	Most students should use this session to answer questions 1, 2 and 3 on pages 64 and 65 of the text book. Those who have already successfully answered the questions could carry on with the **Binary to seven-segment converter** practical. Ask them to revise **Logic Systems** for a formal test next session.

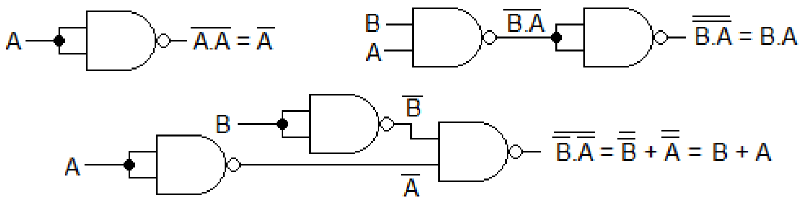
Model Answers

1 (a)

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

(b) $Q = \overline{B.A}$, $Q = \overline{B} + \overline{A}$

(c)



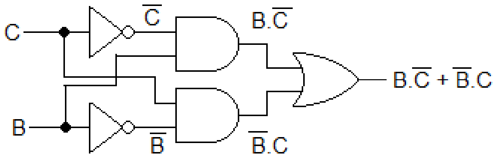
(d) Can reduce the number of integrated circuits required, allow logic systems to take up less space on a circuit board, allow more efficient use of the gates in each integrated circuit and reduce the price of each integrated circuit because more of them are made.

2 (a) From De Morgan's Theorem: $Q = \overline{A}.B.\overline{C} + A.B.\overline{C} + \overline{B}.\overline{\overline{C}}$

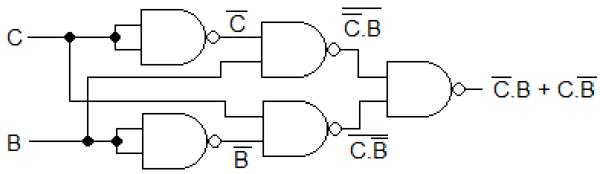
Applying brackets: $Q = B.\overline{C}.(\overline{A} + A) + \overline{B}.C$

Remove contents of brackets: $Q = B.\overline{C} + \overline{B}.C$

(b)



(c)



3 (a) $Q_3 = \overline{B}.\overline{A} + \overline{B}.A + B.\overline{A}$ $Q_2 = \overline{B}.\overline{A} + \overline{B}.A + B.A$

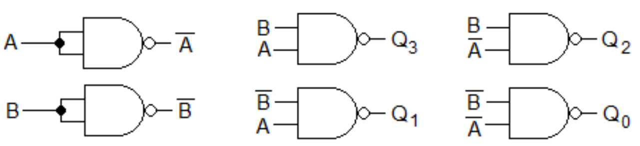
$Q_1 = \overline{B}.\overline{A} + B.\overline{A} + B.A$ $Q_0 = \overline{B}.A + B.\overline{A} + B.A$

(b) Brackets: $Q_2 = \overline{B}.\overline{A} + \overline{B}.A + B.A = \overline{B}.(\overline{A} + A) + B.A = \overline{B} + B.A$

From Race Hazard and Redundancy Theorems: $\overline{B} + B.A = \overline{B} + B.A + A = \overline{B} + A$

From De Morgan's Theorem: $\overline{B.\overline{A}} = \overline{B} + \overline{\overline{A}} = \overline{B} + A$

(c)



(d) Majority of the NOT gate i.c. not used, two different chips required instead of just one.