1.7 Boolean Algebra, Logic Gates and Karnaugh Maps: 30-minute Pack

Total Marks

OCR H446 · Calculators not allowed · about 30 minutes

Name: _____ Date: ____ / ____ / _____

Answer ALL questions. Show all your working.

1 A logic circuit has three inputs, A, B and C. Inputs A and B feed a NAND gate. The output of the NAND gate and input C feed a 2-input OR gate, whose output is Q.

(a) Write the Boolean expression for Q in terms of A, B and C. **(2)**

(b) Complete the truth table for this circuit for all 8 combinations of A, B and C. **(3)**

A B C	A.B	NOT(A.B)	Q
0 0 0	?	?	?
0 0 1	?	?	?
0 1 0	?	?	?
0 1 1	?	?	?
1 0 0	?	?	?
1 0 1	?	?	?
1 1 0	?	?	?
1 1 1	?	?	?

(Total for Question 1 is 5 marks)

---

2 Simplify each of the following Boolean expressions using the laws of Boolean algebra, showing each law you use.

(a) Simplify Q = A.B + A.B' **(2)**

(b) Simplify Q = A + A.B **(2)**

(c) Simplify Q = (A+B).(A+B') **(3)**

(Total for Question 2 is 7 marks)

© Revision Library | revisionlibrary.co.uk

3 This question is about De Morgan's laws.

(a) State De Morgan's first law in Boolean notation. **(1)**

(b) State De Morgan's second law in Boolean notation. **(1)**

(c) Use De Morgan's laws and the laws of Boolean algebra to simplify Q = NOT(NOT A . NOT B) + A.B, showing each step of your working. **(4)**

(Total for Question 3 is 6 marks)

---

4 This question is about the full adder circuit, which has three inputs: A, B and a carry-in, Cin.

(a) Determine the Boolean expression for the Sum output, S, of a full adder in terms of A, B and Cin. **(2)**

(b) Determine the Boolean expression for the Carry-out output, Cout, of a full adder in terms of A, B and Cin. **(2)**

(c) Explain how a full adder can be constructed from two half adders and one OR gate. **(3)**

(d) Complete the truth table for a full adder for all 8 combinations of A, B and Cin. **(4)**

A B Cin	S	Cout
0 0 0	?	?
0 0 1	?	?
0 1 0	?	?
0 1 1	?	?
1 0 0	?	?
1 0 1	?	?
1 1 0	?	?
1 1 1	?	?

(Total for Question 4 is 11 marks)

© Revision Library | revisionlibrary.co.uk

Mark scheme · 1.7 Boolean Algebra, Logic Gates and Karnaugh Maps: 30-minute Pack

Question 1

(a) NAND term correctly represented as NOT(A.B) or (A.B)' **M1**

(a) correct final expression Q = NOT(A.B) + C, oe **A1**

(a) Answer: Q = NOT(A.B) + C

(b) A.B column fully correct: 0,0,0,0,0,0,1,1 **B1**

(b) NOT(A.B) column fully correct: 1,1,1,1,1,1,0,0 **B1**

(b) Q column fully correct: 1,1,1,1,1,1,0,1 **B1**

(b) Answer: Q = 1,1,1,1,1,1,0,1 (only row A=1,B=1,C=0 gives Q=0)

Question 2

(a) factorises correctly: A.(B+B') **M1**

(a) Q = A cao, using B+B'=1 **A1**

(a) Answer: Q = A

(b) factorises correctly: A.(1+B) **M1**

(b) Q = A cao, using 1+B=1 **A1**

(b) Answer: Q = A

(c) applies the distributive law (A+B).(A+C) = A+(B.C) with C=B': Q = A + B.B' **M1**

(c) B.B' = 0, so Q = A + 0 **A1**

(c) Q = A cao **A1**

(c) Answer: Q = A

Question 3

(a) NOT(A.B) = NOT A + NOT B, oe e.g. (A.B)' = A' + B' **B1**

(a) Answer: NOT(A.B) = NOT A + NOT B

(b) NOT(A+B) = NOT A . NOT B, oe e.g. (A+B)' = A'.B' **B1**

(b) Answer: NOT(A+B) = NOT A . NOT B

(c) applies De Morgan's law: NOT(NOT A . NOT B) = A + B **M1**

(c) substitutes to give Q = (A+B) + A.B **M1**

(c) applies absorption law A + A.B = A (with X=A+B, Y=B) to reduce (A+B)+A.B to A+B **M1**

(c) Q = A + B cso **A1**

(c) Answer: Q = A + B

Question 4

(a) recognises S is the XOR of the three inputs, e.g. combining A XOR B first **M1**

(a) S = (A XOR B) XOR Cin, oe A XOR B XOR Cin **A1**

(a) Answer: S = A XOR B XOR Cin

(b) recognises Cout is 1 whenever at least two of the three inputs are 1, expressed as a sum of the pairwise AND terms **M1**

(b) Cout = A.B + B.Cin + A.Cin, oe (e.g. (A.B) + (Cin.(A XOR B))) **A1**

(b) Answer: Cout = A.B + B.Cin + A.Cin

(c) the first half adder takes inputs A and B, producing a partial sum and a partial carry **B1**

(c) the second half adder takes the partial sum and Cin as its two inputs, producing the final Sum output, S, and a second carry **B1**

(c) the two carry outputs (from the first and second half adders) are combined using the OR gate to produce Cout **B1**

(c) Answer: Half adder 1 adds A and B (partial sum + carry1); half adder 2 adds the partial sum to Cin (giving final S + carry2); Cout = carry1 OR carry2.

(d) S correct for rows 1-4: 0, 1, 1, 0 **B1**

(d) S correct for rows 5-8: 1, 0, 0, 1 **B1**

(d) Cout correct for rows 1-4: 0, 0, 0, 1 **B1**

(d) Cout correct for rows 5-8: 0, 1, 1, 1 **B1**

(d) Answer: S = 0,1,1,0,1,0,0,1 and Cout = 0,0,0,1,0,1,1,1

© Revision Library | revisionlibrary.co.uk