A-LEVEL COMPUTER SCIENCE • EXERCISE SHEET

15.2 Boolean Algebra and Logic Circuits

Name: _____ Class: _____ Date: _____ **Total: 31 marks**

KEY WORDS boolean 布尔 • karnaugh map 卡诺图 • truth table 真值表 • boolean algebra 布尔代数 • flip-flop 触发器

Objective

Build the skills to answer exam questions on **Boolean algebra and logic circuits** — simplification, Karnaugh maps, adders and flip-flops.

You must be able to:

- simplify **Boolean** expressions (De Morgan, absorption)
- use a **Karnaugh map** to simplify a truth table
- give the truth tables of a **half/full adder** and describe **SR/JK flip-flops**

1 Worked examples

Study these first. Each one shows the method for a question type used later — follow the steps and you can do the Practice and Exam-style questions yourself.

■ Boolean algebra

`+` = OR, $\cdot$ = AND, overbar = NOT. Useful laws:

- **De Morgan:** $\overline{A+B} = \overline{A} \cdot \overline{B}$ and $\overline{A \cdot B} = \overline{A} + \overline{B}$
- **absorption:** $A + AB = A$
- $A + \overline{A} = 1$, so $Z = AB + A\overline{B} = A(B + \overline{B}) = A$.

Fewer terms → **fewer gates**.

■ Sum-of-products from a truth table

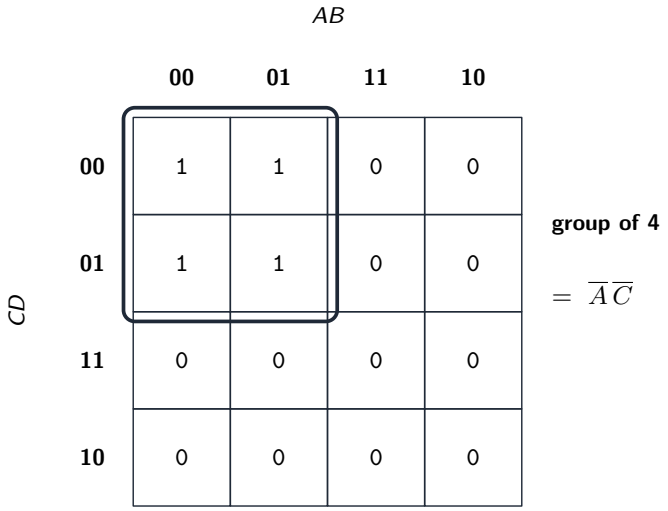
For **each row** where the output is `1`, AND the inputs (a variable as itself if it is 1, or NOT it if 0); then OR these product terms together. If the 1s are at rows $A\overline{B}C$ and ABC:

$$Z = A\overline{B}C + ABC = AC(\overline{B} + B) = AC.$$

This is the **sum-of-products (SOP)** form; simplify it with Boolean algebra or a Karnaugh map.

■ Karnaugh map

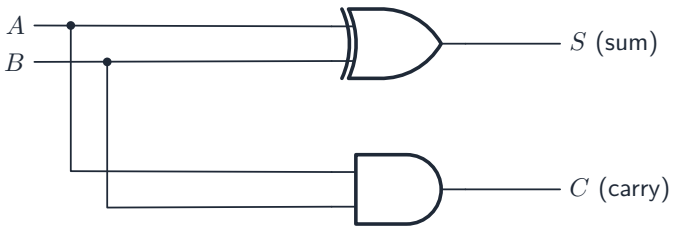
Place each truth-table `1` in the map (Gray-code order), then **group adjacent 1s** in blocks of 1, 2, 4 or 8. A larger group = a simpler term — variables that **change** within the group disappear:



A group of 2 drops 1 variable; a group of 4 drops **2**; a group of 8 drops 3.

■ Half adder and flip-flops

A **half adder** gives $S = A \oplus B$ and $C = A \cdot B$:



A **full adder** also adds a carry-in. A **flip-flop** stores **one bit**: an **SR** flip-flop sets/resets/holds (`S=1,R=1` is invalid); a **JK** uses `1,1` to **toggle**, so it suits counters.

2 Practice

Now apply the methods above.

2.1 Simplify $A + AB$. [1]

2.2 Use **De Morgan's law** to rewrite $\overline{A \cdot B}$. [1]

2.3 State the **sum** and **carry** outputs of a half adder as Boolean expressions. [2]

2.4 State what a **flip-flop** stores. [1]

3 Exam-style questions

3.1 Simplify $Z = AB + A\overline{B}$ using Boolean algebra. Show your steps. [2]

3.2 A K-map has four 1s forming a group where $A = 0$ and $C = 0$ throughout. Write the simplified term, and state how many variables a **group of 4** removes. [2]

3.3 Complete the truth table for a **half adder**. [3]

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

3.4 State **one** difference between an **SR** flip-flop and a **JK** flip-flop. [2]

3.5 A two-input logic circuit gives output `1` only on the rows `A=0, B=1` and `A=1, B=1`. Write the Boolean expression as a **sum-of-products**, then **simplify** it. [3]

3.6 A logic circuit has three inputs A, B and C, and one output X, given by

$$X = \bar{A} \cdot B + A \cdot \bar{B} \cdot C + A \cdot B \cdot C$$

(a) **Complete** the truth table. [4]

A	B	C	X
0	0	0	_____
0	0	1	_____
0	1	0	_____
0	1	1	_____
1	0	0	_____
1	0	1	_____
1	1	0	_____
1	1	1	_____

(b) **Complete** the Karnaugh map for X. [2]

$AB\backslash C$	**0**	**1**
00	_____	_____
01	_____	_____
11	_____	_____
10	_____	_____

(c) **Draw** loops around the optimal groups on your map, and **write** the simplified sum-of-products expression. [4]

(d) **Explain** why the row order of the map is 00, 01, 11, 10 and not 00, 01, 10, 11. [2]

(e) **State** how many two-input gates the simplified expression needs, compared with the original. [2]

Solutions

2.1 $A + AB = A$ (absorption law).

2.2 $\overline{A \cdot B} = \overline{A} + \overline{B}$.

2.3 $S = A \oplus B$ (A XOR B); $C = A \cdot B$ (A AND B).

2.4 One bit (it is bistable — it remembers its state).

3.1 $Z = AB + A\overline{B} = A(B + \overline{B}) = A \cdot 1 = A$.

3.2 Term $= \overline{A}\,\overline{C}$; a group of 4 removes **2** variables.

3.3

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

3.4 In an SR flip-flop the input `S=1, R=1` is **invalid**; a JK flip-flop uses `J=1, K=1` to **toggle** (flip) the output — so JK suits counters.

3.5 SOP: $Z = \overline{A} \cdot B + A \cdot B$; $= B(\overline{A} + A) = B \cdot 1 = B$.

3.6 (a) $X = 0, 0, 1, 1, 0, 1, 0, 1$ for the eight rows in the order given.
(b) Reading the same values into the map: row 00 gives $0, 0$; row 01 gives $1, 1$; row 11 gives $0, 1$; row 10 gives $0, 1$.
(c) Two groups: the whole $AB = 01$ row, which is $\bar{A}B$; and the vertical pair in the $C = 1$ column covering $AB = 11$ and $AB = 10$, which is AC. So $X = \bar{A}B + AC$.
(d) Consecutive rows must differ in **exactly one variable** — that is Gray-code order; it is what makes an adjacent pair of 1s correspond to a term with one variable eliminated, so grouping works at all. With 01 next to 10 two variables change and the grouping would be wrong.
(e) Simplified: one NOT, two ANDs and one OR — **four** gates. The original has three AND terms, two of them three-input, plus two NOTs and two ORs: at least **eight** two-input gates.