Topic 10: Boolean Logic

IGCSE Computer Science 0478 | SenpaiCorner | Mr. Wei

Learning Objectives

By the end of this topic, you will be able to:

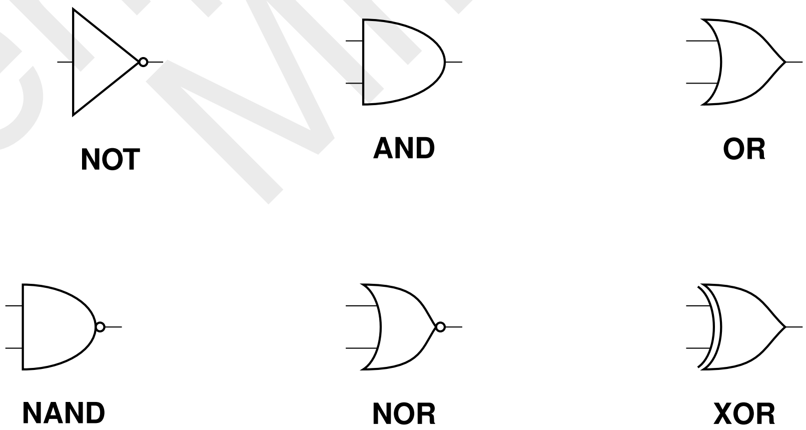
- Identify, draw, and state the function of the six standard logic gates: NOT, AND, OR, NAND, NOR, and XOR.
- Construct and complete truth tables for any combination of the six logic gates.
- Translate between problem statements, logic expressions, and logic circuits.
- Write Boolean algebra expressions from circuits and from truth tables.

10.1 & 10.2: Standard Symbols and Functions of Logic Gates

Computers process data in binary (0s and 1s). Hardware evaluates these bits using logical rules implemented by **logic gates**. Each gate takes one or two binary inputs and produces a single binary output. You must know all six standard gates below.

Gate Circuit Symbols Reference

The diagram below shows the standard ANSI logic gate symbols used in circuit diagrams. Input wires enter from the left; the output wire exits from the right.



Gate Quick Reference

Gate	Expression	Inputs	Output is 1 when...
NOT	$X =$ NOT A	1	A is 0 (always inverts)
AND	$X = A$ AND B	2	**Both** A and B are 1
OR	$X = A$ OR B	2	**At least one** of A, B is 1
NAND	$X = A$ NAND B	2	NOT (both A and B are 1)
NOR	$X = A$ NOR B	2	**Both** A and B are 0
XOR	$X = A$ XOR B	2	A and B are **different**

Truth Tables for All Six Gates

NOT Gate

A	X
0	1
1	0

AND, OR, NAND, NOR, and XOR Gates

A	B	AND	OR	NAND	NOR
0	0	0	0	1	1
0	1	0	1	1	0
1	0	0	1	1	0
1	1	1	1	0	0

A	B	XOR
0	0	0
0	1	1
1	0	1
1	1	0

Quick Memory Guide for Gate Outputs

- **AND**: Both must be 1 to get 1.
- **OR**: At least one must be 1 to get 1.
- **NOT**: Flips the input.
- **NAND** / **NOR**: Apply the AND / OR rule, then flip the result.
- **XOR**: Different inputs give 1; same inputs give 0.

Confusing NAND with NOT-A AND NOT-B

NAND means NOT (A AND B) — it is NOT the same as (NOT A) AND (NOT B). Always apply the NOT *after* the AND or OR operation, never to each input individually.

10.3: Applying Boolean Logic

This section requires translating between three representations of Boolean logic: **problem statements**, **logic expressions**, and **logic circuits / truth tables**.

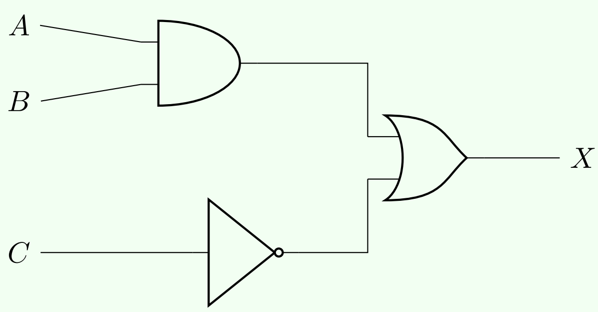
10.3(a): Creating Logic Circuits

You must be able to draw logic circuits based on:

1. **Problem statements** — identify the conditions (inputs) and the outcome (output), then map connective words to gate types.
2. **Logic expressions** — map expressions directly to gates. Evaluate brackets first (like BODMAS in standard mathematics). Example: $X = (A \text{ AND } B) \text{ OR } (\text{NOT } C)$
3. **Truth tables** — identify rows where the output is 1, then construct gates to isolate those input combinations.

Circuit from Expression

Draw the circuit for $X = (A \text{ AND } B) \text{ OR } (\text{NOT } C)$.



Reading the circuit: A and B feed the AND gate, producing $(A \text{ AND } B)$. C feeds the NOT gate, producing $(\text{NOT } C)$. Both results feed the OR gate, giving the final output X.

10.3(b): Completing Truth Tables

For complex circuits, add **intermediate columns** to track the output of each gate as data passes through the circuit.

1. List all possible input combinations. For n inputs there are 2^n rows (3 inputs $= 8$ rows; 2 inputs $= 4$ rows).
2. Work from the innermost brackets outward, adding one column per intermediate gate.
3. Fill the final output column last.

Worked example: $X = A \text{ AND } (B \text{ OR } C)$

A	B	C	*Intermediate:* $(B$ *OR* $C)$	*Final:* $X = A$ *AND* $(B$ *OR* $C)$
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	1	0
1	0	0	0	0
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

Show Intermediate Columns in the Exam

Always show intermediate columns in exam answers. Examiners award method marks for correctly completed intermediate columns even if the final output column contains errors.

Missing Rows in a Truth Table

For n inputs a truth table requires exactly 2^n rows. With 3 inputs that is $2^3 = 8$ rows. Students who list only 6 rows miss valid input combinations and lose all marks for those rows. Count your rows before writing the final output column.

10.3(c): Writing Logic Expressions

Expression from a Problem Statement

"The alarm (X) rings if the door is open (A) AND the system is armed (B), OR if the panic button is pressed (C)."

Translating directly:

$$X = (A \text{ AND } B) \text{ OR } C$$

Deriving a Logic Expression From...	**Method / Steps**
A circuit	Work backwards from the output gate. Identify the inputs to each gate and trace back to the original inputs A, B, C. Group each intermediate result in parentheses.
A problem statement	Assign a letter to each condition, then translate connective words into operators: • "and" $\rightarrow$ AND • "or" $\rightarrow$ OR • "not" / "unless" / "only if not" $\rightarrow$ NOT
A truth table	1. Locate every row where the output is 1. 2. For each such row, write an AND expression for the inputs (use NOT for any 0-valued input in that row). 3. Join all such AND expressions together with OR operators.

Ambiguous Expressions Without Brackets

In Boolean algebra, AND takes precedence over OR (like multiplication before addition). Always use parentheses to make the intended grouping explicit. Writing $X = A$ AND B OR C is ambiguous — write $X = (A$ AND $B)$ OR C to make the order of evaluation clear. Cambridge mark schemes penalise missing brackets.

Writing an Expression from a Truth Table

Task: The truth table below has output 1 in exactly two rows. Write the Boolean expression for X.

A	B	C	X
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	**1**
1	1	0	**1**
1	1	1	0

Step 1 — Identify output-1 rows: Row (1,0,1): $A = 1$, $B = 0$, $C = 1$. Row (1,1,0): $A = 1$, $B = 1$, $C = 0$.

Step 2 — Write AND clause for each row:

- Row (1,0,1): A AND (NOT B) AND C
- Row (1,1,0): A AND B AND (NOT C)

Step 3 — Join clauses with OR:

$$X = \big(A \text{ AND } (\text{NOT } B) \text{ AND } C\big) \text{ OR } \big(A \text{ AND } B \text{ AND } (\text{NOT } C)\big)$$

Notice that both output-1 rows have $A = 1$ and exactly one of B, C as 1 — this is the XOR relationship between B and C, but always express it by tracing from the truth table unless you are certain of the simplified form.