LADDER LOGIC 2

🤖 Chapter 5-2: Basic Instructions in Ladder Logic

Boolean Logic and the DNA of Control

Now that you know what Ladder Logic looks like, it's time to understand **how it actually decides things**. And that takes us straight into **Boolean logic** — the bedrock of all decision-making in PLCs.

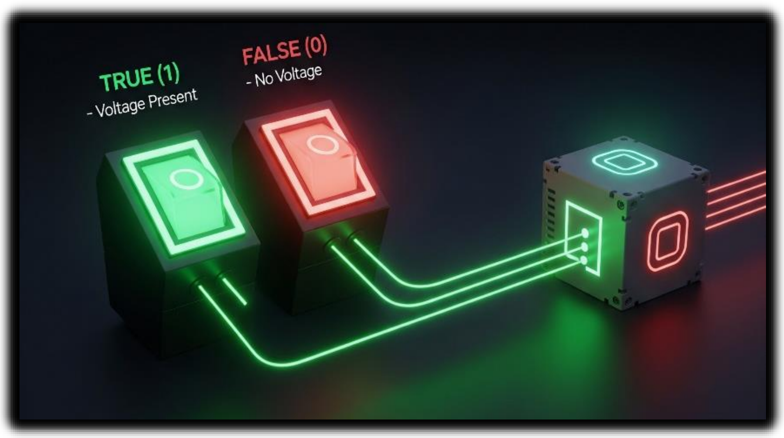
Don't worry, this isn't Digital Systems 101 — we're not breaking out Karnaugh maps or Boolean algebra proofs. We're just gonna look at the essentials: **AND** and **OR** logic.

🔲 Boolean Logic in Plain Language

PLC logic is built on simple True/False decisions — like light switches:

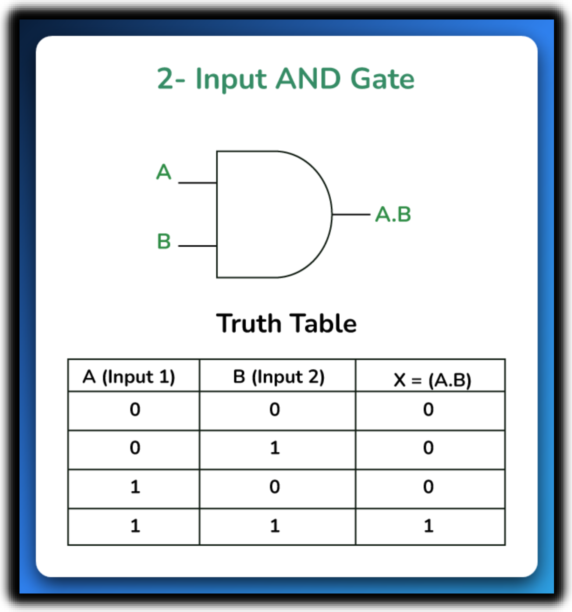
- **TRUE (1)** → There's voltage or a condition is satisfied
- **FALSE (0)** → No voltage or the condition isn't satisfied

From this binary setup, we build logic gates. In Ladder Logic, these gates are **not separate components** — they're created through **how you arrange your rungs and contacts**.



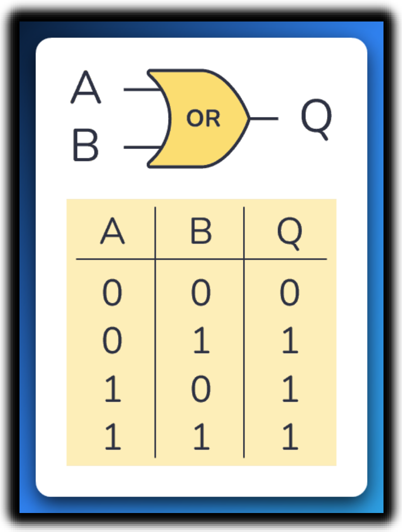
🔗 The AND Gate — Series Logic

AND operations are analogous to **multiplication.**



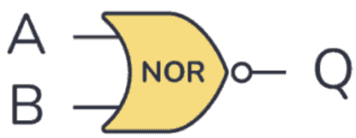
A (Input 1)	B (Input 2)	X = (A.B)
0	0	0
0	1	0
1	0	0
1	1	1

OR operations are comparable to **addition**.



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

NOR gate(no need to memorize this):



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

XOR gate(no need to memorize this): A digital logic gate that outputs a "true" (or "1") signal only when an odd number of its inputs are "true" (or "1")

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

🔌 AND gate Ladder Implementation:

Use **normally open contacts in series**.

This is saying:

*“Only if A **AND** B are both TRUE, then turn ON the output.”*

If either A or B is FALSE, current stops flowing — just like if one switch in a series circuit is off.

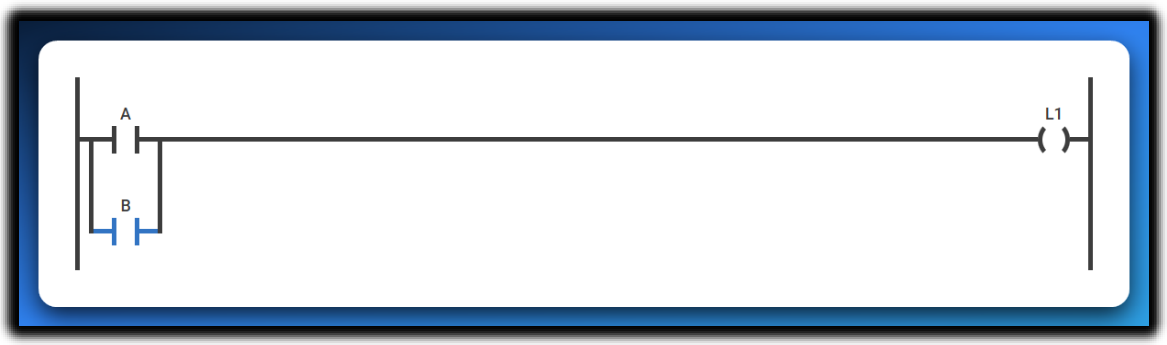
🔌 OR gate Ladder Implementation:

Use **normally open contacts in parallel**.

This is saying:

“If A **OR** B is TRUE, then turn ON the output.”

As long as at least one of them is ON, the rung is complete and current flows to the output.



If **A is TRUE** (and B is FALSE), the path through A "closes," allowing logical power to flow to the output.

If **B is TRUE** (and A is FALSE), the path through B "closes," allowing logical power to flow to the output.

If **both A and B are TRUE**, both paths "close," and logical power still flows to the output.

Only if **both A and B are FALSE** will both paths remain "open," stopping the logical power flow and keeping the output OFF.

💡 Key Concept: Contacts Are Logic Conditions

In Ladder Logic, your contacts (—| |—) aren't checking voltage directly — they're checking **bit states** in memory.

- A contact in a rung is **like a mini IF-statement**.
- When you place them **in series**, it means "all must be true" (AND).
- When you place them **in parallel**, it means "any can be true" (OR).

And these logic structures determine whether **your output coils (—( )—)** get energized or not.

🧠 Analogy Time:

- Think of series logic like **security clearance**:
  "You need both a keycard **and** a password to enter."
- Think of parallel logic like **alarm triggers**:
  "If **any** of the sensors trip, sound the alarm."

📌 Summary:

- **AND Logic = Series contacts** → All must be TRUE.
- **OR Logic = Parallel contacts** → At least one must be TRUE.
- No separate gate components — it's all about **how you arrange your contacts** in the rung.
