Lesson 4: Logic Gates & Boolean Algebra

1. Logic Gates

Gate Lexicon

Gate	Expression	Output is 1 when...
NOT	$L = \overline{A}$	input is 0.
AND	$L = A \cdot B$	all inputs are 1.
OR	$L = A + B$	at least one input is 1.
NAND	$L = \overline{A \cdot B}$	any input is 0.
NOR	$L = \overline{A + B}$	all inputs are 0.
XOR	$L = A \oplus B$	inputs are different.
XNOR	$L = \overline{A \oplus B}$	inputs are the same.

Universal Gates (NAND & NOR)

- **Why universal?** Any logic gate can be constructed using only NAND gates or only NOR gates. This makes manufacturing ICs cheaper and simpler.

Implementations with NAND

- **NOT A** = $(A \cdot A)'$
- **A AND B** = $((A \cdot B)')'$
- **A OR B** = $(A' \cdot B')'$

2. Boolean Algebra

Key Laws & Theorems

- **Identity:** $A + 0 = A$, $A \cdot 1 = A$
- **Complement:** $A + \overline{A} = 1$, $A \cdot \overline{A} = 0$
- **Commutative:** $A + B = B + A$
- **Associative:** $A + (B + C) = (A + B) + C$
- **Distributive:** $A(B + C) = AB + AC$
- **De Morgan's:**
  - $\overline{A \cdot B} = \overline{A} + \overline{B}$
  - $\overline{A + B} = \overline{A} \cdot \overline{B}$
- **Redundancy:** $A + \overline{A}B = A + B$

3. Standard Forms

SOP & POS

- **SOP (Sum of Products):** Sum of AND terms (Minterms). Derived from rows where output is **1**.
  - *Ex:* $F = \overline{A}BC + A\overline{B}C$
- **POS (Product of Sums):** Product of OR terms (Maxterms). Derived from rows where output is **0**.
  - *Ex:* $F = (A + B + C)(A + \overline{B} + C)$

Deriving SOP from a Truth Table

A	B	C	F	Minterm
0	0	0	0	
0	0	1	**1**	$\rightarrow \overline{AB}C$
0	1	0	0	
0	1	1	**1**	$\rightarrow \overline{A}BC$
...	...	...	...	

SOP Expression: $F = \overline{AB}C + \overline{A}BC$

4. Karnaugh Maps (K-Maps)

Simplification Rules

1. Group only 1s, never 0s.
2. Groups must be rectangular & contain a power of 2 number of cells (1, 2, 4, 8...).
3. No diagonal groups.
4. Make groups as large as possible.
5. Groups can overlap. Wrap-around is allowed.
6. Use the fewest number of groups possible.

3-Variable K-Map Example

Simplify $F = \overline{AB}C + \overline{A}BC + A\overline{BC} + A\overline{B}C$

1. **Map the 1s:**

	AB			
C	00	01	11	10
0				**1**
1	**1**	**1**		**1**

2. **Group the 1s:** Two groups are made.
   - Group 1 (Green): The two 1s in column AB=10. This simplifies to $A\overline{B}$.
   - Group 2 (Blue): The two 1s in row C=1, columns AB=00 and AB=01. This simplifies to $\overline{A}C$.
3. **Final Expression:** $F = A\overline{B} + \overline{A}C$

5. Circuit Types

Combinational vs. Sequential

- **Combinational Circuits:**
  - Output depends **only** on the current inputs.
  - Has **no memory**.
  - *Examples: Adders, Decoders.*
- **Sequential Circuits:**
  - Output depends on current inputs **and** past states.
  - Has **memory** to store past states.
  - Uses a **feedback loop** (output is fed back as an input).
  - *Examples: Flip-Flops, Counters.*

6. Practical Circuits

Adders

- **Half Adder:** Adds 2 bits.
  - **Sum** $= A \oplus B$
  - **Carry** $= A \cdot B$
- **Full Adder:** Adds 3 bits (A, B, C_{in}). Built with two Half Adders and an OR gate.
  - **Sum** $= A \oplus B \oplus C_{in}$
  - **Carry Out** $= A \cdot B + C_{in}(A \oplus B)$

Memory

- **Flip-Flop:** The basic building block of memory. A sequential circuit that can **store** a single bit (0 or 1).
- **RS Latch (Flip-Flop):** Simplest type, built with cross-coupled NAND or NOR gates. Has Set (S) and Reset (R) inputs.

Design Example: Simple Alarm

Problem: An alarm (F) should sound if a sensor (A) is triggered, but only if the system is armed (B) and the door is closed (C).

1. **Truth Table:** The only time F=1 is when A=1, B=1, and C=1.
2. **Expression:** The SOP expression is simply the minterm for that one case: $F = A \cdot B \cdot C$.
3. **Circuit:** A single 3-input AND gate.