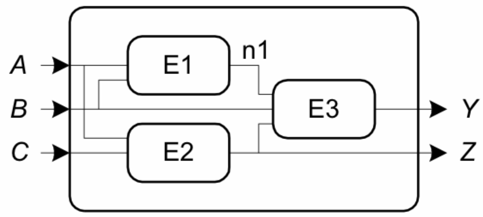
# Logic Circuits

↳ composed of

inputs → | functional spec. / timing spec. | → outputs

↳ high-level classification.

- **Nodes**
  - Inputs: *A*, *B*, *C*
  - Outputs: *Y*, *Z*
  - Internal: n1
- **Circuit elements**
  - E1, E2, E3
  - Each a circuit



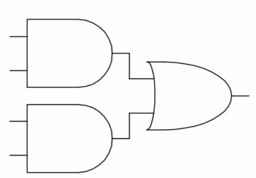

↳ types of logic circuits.

→ combinational → no memory
→ output depends on current val of inputs.

→ sequential → memory
→ outputs det. on current & prev inputs.

## Rules of Combinational Composition

- Every element is combinational
- Every node is either an input or connects to *exactly one* output
- The circuit contains no cyclic paths
- **Example:**

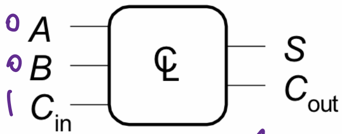


## Boolean Equations

this is an adder

- Functional specification of outputs in terms of inputs
- **Example:** $S = F(A, B, C_{in})$
  $C_{out} = F(A, B, C_{in})$

0+1=1




$S = \overset{0}{A} \oplus \overset{0}{B} \oplus \overset{1}{C_{in}} = 1.$

$C_{out} = AB + AC_{in} + BC_{in}$

$\underset{2^1\,2^0}{01}{}_2 = 1.$

0 0 0



## Some Definitions

- variable: A, B, C... (without bars)
- **Complement:** variable with a bar over it
  $\bar{A}, \bar{B}, \bar{C}$
- **Literal:** variable or its complement
  $A, \bar{A}, B, \bar{B}, C, \bar{C}$
- **Implicant:** product of literals
  $AB\bar{C}, \bar{A}C, BC$
- **Minterm:** product that includes all input variables
  $AB\bar{C}, A\bar{B}\bar{C}, ABC$
- **Maxterm:** sum that includes all input variables
  $(A+\bar{B}+C), (\bar{A}+B+\bar{C}), (\bar{A}+\bar{B}+C)$

# Sum-of-Products (SOP) Form

- All equations can be written in SOP form
- Each row has a **minterm**
- A minterm is a **product** (AND) of literals
- Each minterm is **TRUE** for that row (and only that row)
- Form function by **ORing minterms** where **output is 1**
- Thus, a **sum** (OR) of **products** (AND terms)

| A | B | Y | minterm | minterm name |
|---|---|---|---|---|
| 0 | 0 | 0 | $\overline{A}\ \overline{B}$ | $m_0$ |
| 0 | 1 | **1** | $\overline{A}\ B$ | $m_1$ |
| 1 | 0 | 0 | $A\ \overline{B}$ | $m_2$ |
| 1 | 1 | **1** | $A\ B$ | $m_3$ |

$$Y = F(A, B) = \underbrace{\underbrace{\overline{A}B}_{\text{AND}} + \underbrace{AB}_{\text{AND}}}_{\text{OR}} = \Sigma(1, 3)$$



A = 1

we are defining A differently because we want the default to give us the desired outcome.

A = 0

# Product-of-Sums (POS) Form

- All Boolean equations can be written in POS form
- Each row has a **maxterm**
- A maxterm is a **sum** (OR) of literals
- Each maxterm is **FALSE** for that row (and only that row)
- Form function by **ANDing maxterms** where **output is 0**
- Thus, a **product** (AND) of **sums** (OR terms)

| A | B | Y | maxterm | maxterm name |
|---|---|---|---|---|
| 0 | 0 | 0 | $A + B$ | $M_0$ |
| 0 | 1 | 1 | $A + \overline{B}$ | $M_1$ |
| 1 | 0 | 0 | $\overline{A} + B$ | $M_2$ |
| 1 | 1 | 1 | $\overline{A} + \overline{B}$ | $M_3$ |

$$Y = F(A, B) = (A + B)(\overline{A} + B) = \Pi(0, 2)$$



The 2 forms are related $\overline{\overline{A}B + AB} = (A+B)(\overline{A}+B)$

An example.

## SOP & POS Form

- **SOP** – sum-of-products

| O | C | E | minterm |
|---|---|---|---|
| 0 | 0 | 0 | $\overline{O}\ \overline{C}$ |
| 0 | 1 | 0 | $\overline{O}\ C$ |
| 1 | 0 | 1 | $O\ \overline{C}$ |
| 1 | 1 | 0 | $O\ C$ |

$$E = O\overline{C}$$
$$= \Sigma(2)$$

- **POS** – product-of-sums

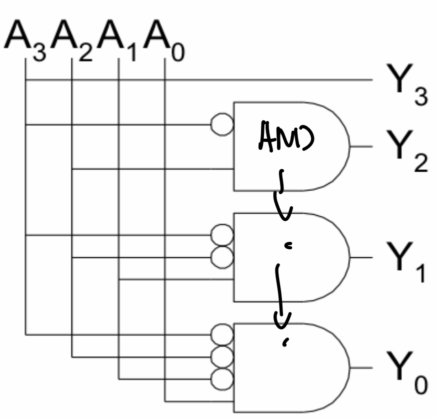
| O | C | E | maxterm |
|---|---|---|---|
| 0 | 0 | 0 | $O + C$ |
| 0 | 1 | 0 | $O + \overline{C}$ |
| 1 | 0 | 1 | $\overline{O} + C$ |
| 1 | 1 | 0 | $\overline{O} + \overline{C}$ |

$$E = (O + C)(O + \overline{C})(\overline{O} + \overline{C})$$
$$= \Pi(0, 1, 3)$$

An example of multiple-output circuit.

## Priority Circuit Hardware

| $A_3$ | $A_2$ | $A_1$ | $A_0$ | $Y_3$ | $Y_2$ | $Y_1$ | $Y_0$ |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |



## X-Notation:

# Don't Cares: X

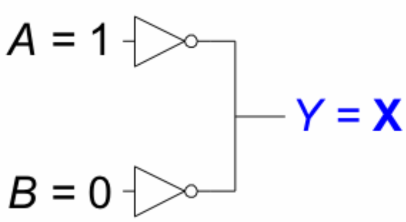
| $A_3$ | $A_2$ | $A_1$ | $A_0$ | $Y_3$ | $Y_2$ | $Y_1$ | $Y_0$ |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |

| $A_3$ | $A_2$ | $A_1$ | $A_0$ | $Y_3$ | $Y_2$ | $Y_1$ | $Y$ |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | X | 0 | 0 | 1 | 0 |
| 0 | 1 | X | X | 0 | 1 | 0 | 0 |
| 1 | X | X | X | 1 | 0 | 0 | 0 |



# Contention: X

- Contention: circuit tries to drive output to 1 **and** 0
  - Actual value somewhere in between
  - Could be 0, 1, or in forbidden zone
  - Might change with voltage, temperature, time, noise
  - Often causes excessive power dissipation



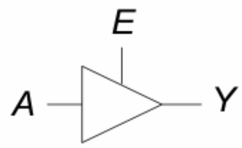

- **Warnings:**
  - Contention usually indicates a **bug**.
  - **X is used for**
    - **"don't care" in a truth table**
    - **and contention in logic simulation**

Floating: Z

# Floating: Z

- Floating, high impedance, open, high Z
- Floating output might be 0, 1, or somewhere in between
  - A voltmeter won't indicate whether a node is floating

**Tristate Buffer**



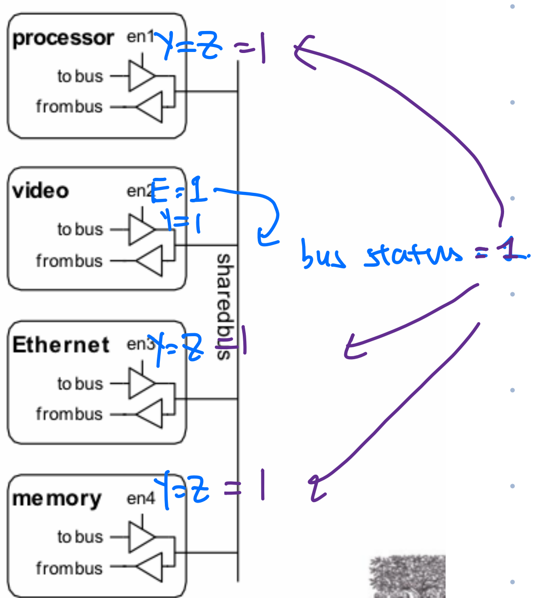

| *E* | *A* | *Y* |
|---|---|---|
| 0 | 0 | Z |
| 0 | 1 | Z |
| 1 | 0 | 0 |
| 1 | 1 | 1 |



# Tristate Busses

- Floating nodes are used in tristate busses
  - Many different drivers
  - Exactly one is active at once

Z's value is set by the bus's state.






# Boolean Algebra

# Boolean Axioms

## Boolean Axioms

| Number | Axiom | Dual | Name |
|---|---|---|---|
| A1 | B = 0 if B ≠ 1 | B = 1 if B ≠ 0 | Binary Field |
| A2 | $0 = \overline{1}$ | $1 = \overline{0}$ | NOT |
| A3 | 0 • 0 = 0 | 1 + 1 = 1 | AND/OR |
| A4 | 1 • 1 = 1 | 0 + 0 = 0 | AND/OR |
| A5 | 0 • 1 = 1 • 0 = 0 | 1 + 0 = 0 + 1 = 1 | AND/OR |

**Dual:** Replace: • with +
0 with 1



## Boolean Theorems of One Variable

| Number | Theorem | Name |
|---|---|---|
| T1 | B • 1 = B | Identity |
| T2 | B • 0 = 0 | Null Element |
| T3 | B • B = B | Idempotency |
| T4 | $\overline{\overline{B}} = B$ | Involution |
| T5 | $B \bullet \overline{B} = 0$ | Complements |

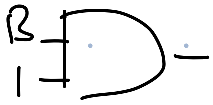


## Boolean Theorems of One Variable

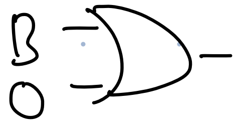
| Number | Theorem | Dual | Name |
|---|---|---|---|
| T1 | B • 1 = B | B + 0 = B | Identity |
| T2 | B • 0 = 0 | B + 1 = 1 | Null Element |
| T3 | B • B = B | B + B = B | Idempotency |
| T4 | $\overline{\overline{B}} = B$ | | Involution |
| T5 | $B \bullet \overline{B} = 0$ | $B + \overline{B} = 1$ | Complements |

**Dual:** Replace: • with +
0 with 1

## T1: Identity Theorem



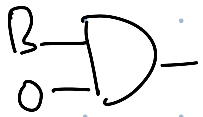

$= B$ ,



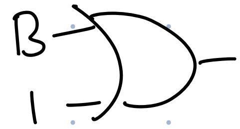

$= B$

$B \cdot 1 = B$ $\qquad$ $B + 0 = B$

## T2: Null Element Theorem.



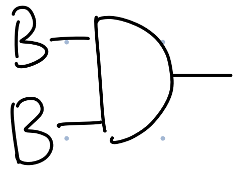

$= 0$



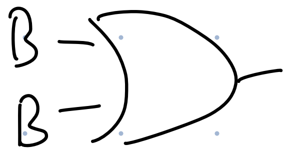

$= 1$

$B \cdot 0 = B$ $\qquad$ $B + 1 = 1$

## T3: Idempotency Theorem




$= B$

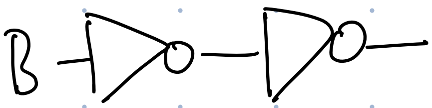



$= B$

$B \cdot B = B$ $\qquad$ $B + B = B$

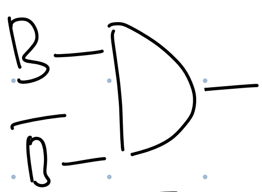
## T4: Identity Theorem

$\overline{\overline{B}} = B$

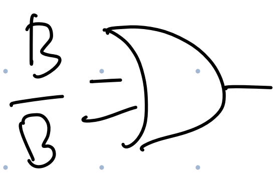



$= B$

## T5: Complement Theorem




$= 0$




$= 1$

$B \cdot \overline{B} = 0$ $\qquad$ $B + \overline{B} = 1$