A Hello Email to Your Grader

- If you haven't done so, please send a Hello email to your grader with a proof of CS 237 or an equivalent (e.g., Computer Organization, Digital Logic, Assembly Language, etc.)

- Tony La: tonyla858@gmail.com

Lab Assingment1 is available now!

- Get familiar with LogicWorks5!
- It will be due by 11:59 pm on Feb. 16 Friday.
- It is an individual assignment.
- You do not need to buy the LogicWorks book.

Logic and Computer Design Fundamentals

Chapter 2 – Combinational Logic Circuits

Part 1 – Gate Circuits and Boolean Equations

Charles Kime & Thomas Kaminski

Overview

- Part 1 – Gate Circuits and Boolean Equations
  - Binary Logic and Gates
  - Boolean Algebra
  - Standard Forms
- Part 2 – Circuit Optimization
  - Two-Level Optimization
  - Map Manipulation
  - Multi-Level Circuit Optimization
- Part 3 – Additional Gates and Circuits
  - Other Gate Types
  - Exclusive-OR Operator and Gates
  - High-Impedance Outputs

Binary Logic and Gates

- <u>Binary variables</u> take on one of two values.
- <u>Logical operators</u> operate on binary values and binary variables.
- Basic logical operators are the <u>logic functions</u> AND, OR and NOT.
- <u>Logic gates</u> implement logic functions.
- <u>Boolean Algebra</u>: a useful mathematical system for specifying and transforming logic functions.
- We study Boolean algebra as foundation for designing and analyzing digital systems!

Binary Variables

- **Recall that the two binary values have different names:**
  - **True/False**
  - **On/Off**
  - **Yes/No**
  - **1/0**
- **We use 1 and 0 to denote the two values.**
- **Variable identifier examples:**
  - **A, B, y, z, or X_1 for now**
  - **RESET, START_IT, or ADD1 later**

Logical Operations

- The three basic logical operations are:
  - AND
  - OR
  - NOT
- AND is denoted by a dot (·).
- OR is denoted by a plus (+).
- NOT is denoted by an overbar (¯), a single quote mark (') after, or (~) before the variable.

Notation Examples

- **Examples:**
  - $Y = A \cdot B$ **is read "Y is equal to A AND B."**
  - $z = x + y$ **is read "z is equal to x OR y."**
  - $X = \overline{A}$ **is read "X is equal to NOT A."**
- **Note: The statement:**

1 + 1 = 2 (read "one <u>plus</u> one equals two")

is not the same as

1 + 1 = 1 (read "1 <u>or</u> 1 equals 1").

Operator Definitions

- **Operations are defined on the values "0" and "1" for each operator:**

AND	OR	NOT
$0 \cdot 0 = 0$	$0 + 0 = 0$	$\overline{0} = 1$
$0 \cdot 1 = 0$	$0 + 1 = 1$	$\overline{1} = 0$
$1 \cdot 0 = 0$	$1 + 0 = 1$	
$1 \cdot 1 = 1$	$1 + 1 = 1$	

Truth Tables

- ***Truth table*** **– a tabular listing of the values of a function for all possible combinations of values on its arguments**
- **Example: Truth tables for the basic logic operations:**

AND		
X	Y	$Z = X \cdot Y$
0	0	0
0	1	0
1	0	0
1	1	1

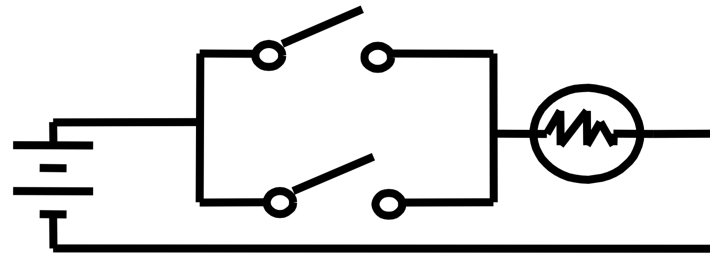
OR		
X	Y	$Z = X + Y$
0	0	0
0	1	1
1	0	1
1	1	1

NOT	
X	$Z = \overline{X}$
0	1
1	0

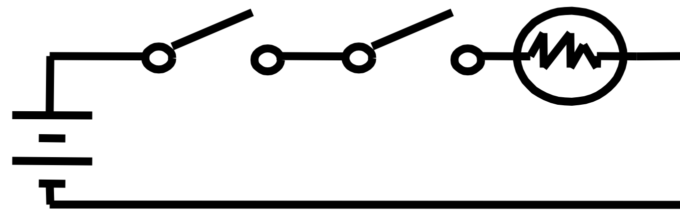
Logic Function Implementation

- **Using Switches**
  - **For inputs:**
    - **logic 1 is switch closed**
    - **logic 0 is switch open**
  - **For outputs:**
    - **logic 1 is light on**
    - **logic 0 is light off.**
  - **NOT uses a switch such that:**
    - **logic 1 is switch open**
    - **logic 0 is switch closed**

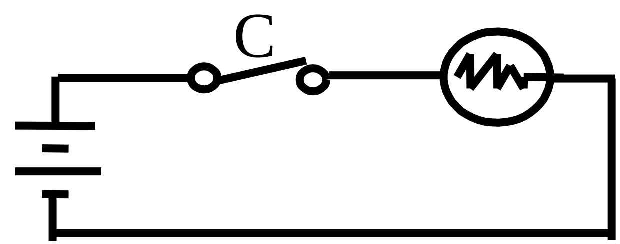
Switches in parallel => OR



Switches in series => AND

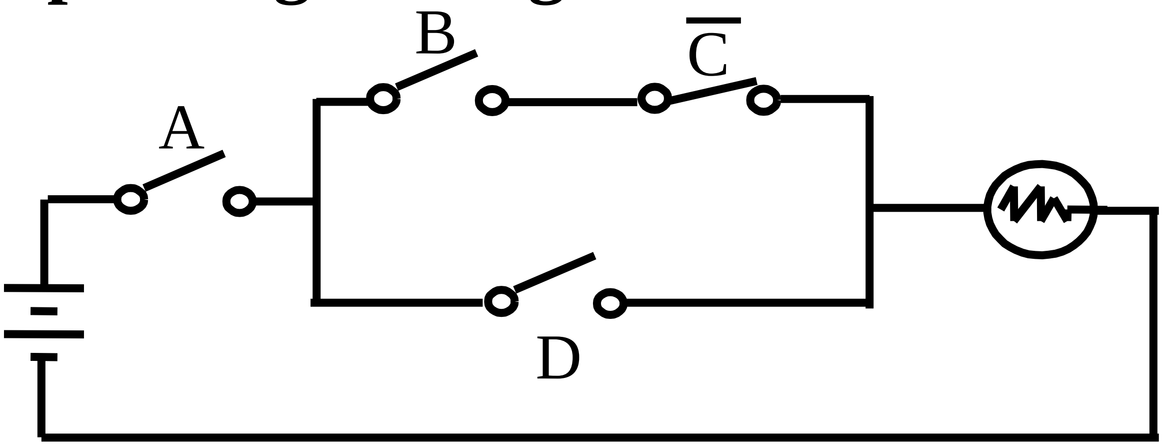


Normally-closed switch => NOT



Logic Function Implementation (Continued)

- **Example: Logic Using Switches**



- **Light is on (L = 1) for**

  L(A, B, C, D) =

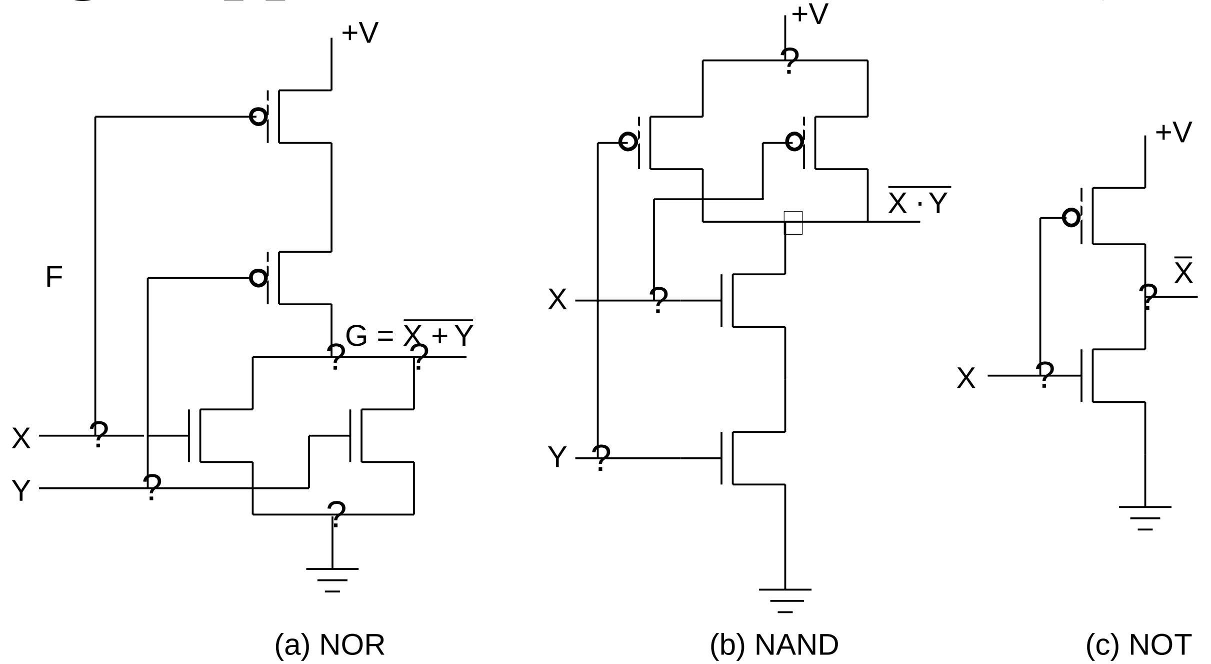
  and off (L = 0), otherwise.

Logic Gates

- In the earliest computers, switches were opened and closed by magnetic fields produced by energizing coils in *relays*. The switches in turn opened and closed the current paths.
- Later, *vacuum tubes* that open and close current paths electronically replaced relays.
- Today, *transistors* are used as electronic switches that open and close current paths.

Logic Gates (continued)

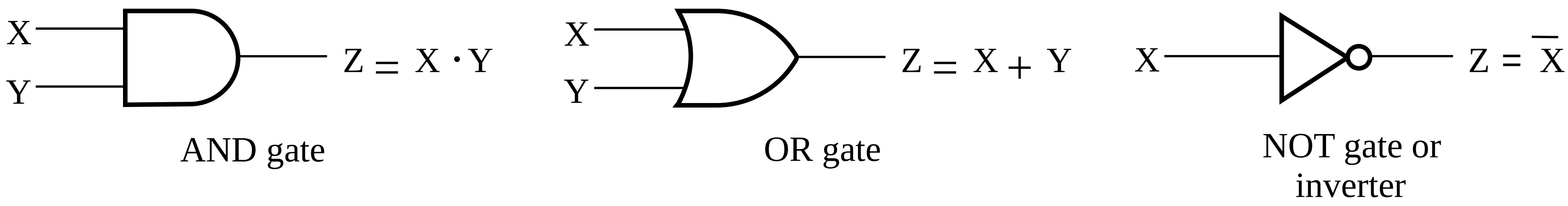
- **Implementation of logic gates with transistors (See Reading Supplement – CMOS Circuits)**



- **Transistor or tube implementations of logic functions are called <u>logic gates</u> or just <u>gates</u>**
- **Transistor gate circuits can be modeled by switch circuits**

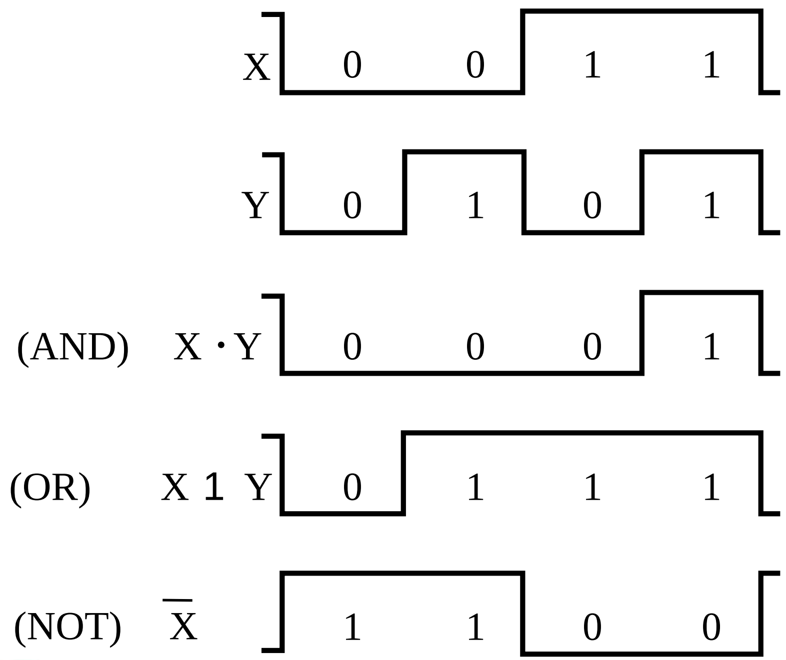
Logic Gate Symbols and Behavior

- **Logic gates have special symbols:**



(a) Graphic symbols

- **And waveform behavior in time as follows:**



(b) Timing diagram

Logic Diagrams and Expressions

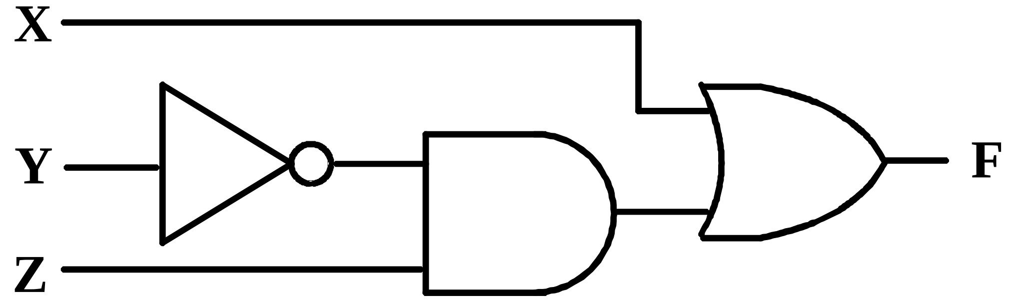
Truth Table

X Y Z	$F = X + \overline{Y} \cdot Z$
0 0 0	0
0 0 1	1
0 1 0	0
0 1 1	0
1 0 0	1
1 0 1	1
1 1 0	1
1 1 1	1

Equation

$$F = X + \overline{Y}\,Z$$

Logic Diagram



- **Boolean equations, truth tables and logic diagrams describe the same function!**
- **Truth tables are unique; expressions and logic diagrams are not. This gives flexibility in implementing functions.**

Boolean Algebra

- **An algebraic structure defined on a set of at least two elements, together with three binary operators (denoted +, · and $\overline{\ }$) that satisfies the following basic identities:**

1.	$X + 0 = X$	2.	$X \cdot 1 = X$	
3.	$X + 1 = 1$	4.	$X \cdot 0 = 0$	
5.	$X + X = X$	6.	$X \cdot X = X$	
7.	$X + \overline{X} = 1$	8.	$X \cdot \overline{X} = 0$	
9.	$\overline{\overline{X}} = X$			
10.	$X + Y = Y + X$	11.	$XY = YX$	Commutative
12.	$(X + Y) + Z = X + (Y + Z)$	13.	$(XY)Z = X(YZ)$	Associative
14.	$X(Y + Z) = XY + XZ$	15.	$X + YZ = (X + Y)(X + Z)$	Distributive
16.	$\overline{X + Y} = \overline{X} \cdot \overline{Y}$	17.	$\overline{X \cdot Y} = \overline{X} + \overline{Y}$	DeMorgan's

Some Properties of Identities & the Algebra

- **If the meaning is unambiguous, we leave out the symbol "·"**
- **The identities above are organized into pairs. These pairs have names as follows:**

  1-4 Existence of 0 and 1 — **5-6 Idempotence**

  7-8 Existence of complement — **9 Involution**

  10-11 Commutative Laws — **12-13 Associative Laws**

  14-15 Distributive Laws — **16-17 DeMorgan's Laws**

- **The <u>dual</u> of an algebraic expression is obtained by interchanging + and · and interchanging 0's and 1's.**
- **The identities appear in <u>dual</u> pairs. When there is only one identity on a line the identity is <u>self-dual</u>, i. e., the dual expression = the original expression.**

Some Properties of Identities & the Algebra (Continued)

- **Unless it happens to be self-dual, the dual of an expression does not equal the expression itself.**
- **Example:** $F = (A + \overline{C}) \cdot B + 0$

  dual $F = (A \cdot \overline{C} + B) \cdot 1 = A \cdot \overline{C} + B$
- **Example:** $G = X \cdot Y + (\overline{W + Z})$

  dual $G =$
- **Example:** $H = A \cdot B + A \cdot C + B \cdot C$

  dual $H =$
- **Are any of these functions self-dual?**

Boolean Operator Precedence

- **The order of evaluation in a Boolean expression is:**
  1. Parentheses
  2. NOT
  3. AND
  4. OR
- **Consequence: Parentheses appear around OR expressions**
- **Example:** $F = A(B + C)(C + \overline{D})$

Example 1: Boolean Algebraic Proof

- $A + A \cdot B = A$ (Absorption Theorem)

Proof Steps	Justification (identity or theorem)
$A + A \cdot B$	
$= A \cdot 1 + A \cdot B$	$X = X \cdot 1$
$= A \cdot (1 + B)$	$X \cdot Y + X \cdot Z = X \cdot (Y + Z)$ (Distributive Law)
$= A \cdot 1$	$1 + X = 1$
$= A$	$X \cdot 1 = X$

- **Our primary reason for doing proofs is to learn:**
  - **Careful and efficient use of the identities and theorems of Boolean algebra, and**
  - **How to choose the appropriate identity or theorem to apply to make forward progress, irrespective of the application.**

Example 2: Boolean Algebraic Proofs

- $AB + \overline{A}C + BC = AB + \overline{A}C$ **(Consensus Theorem)**

Proof Steps **Justification (identity or theorem)**

$AB + \overline{A}C + BC$

$= AB + \overline{A}C + 1 \cdot BC$ **?**

$= AB + \overline{A}C + (A + \overline{A}) \cdot BC$ **?**

$=$

Example 3: Boolean Algebraic Proofs

- $(\overline{X+Y})Z + X\overline{Y} = \overline{Y}(X+Z)$

Proof Steps **Justification (identity or theorem)**

$(\overline{X+Y})Z + X\overline{Y}$

$=$

Useful Theorems

- $x \cdot y + \overline{x} \cdot y = y \quad (x + y)(\overline{x} + y) = y$ **Minimization**
- $x + x \cdot y = x \quad x \cdot (x + y) = x$ **Absorption**
- $x + \overline{x} \cdot y = x + y \quad x \cdot (\overline{x} + y) = x \cdot y$ **Simplification**
- $x \cdot y + \overline{x} \cdot z + y \cdot z = x \cdot y + \overline{x} \cdot z$ **Consensus**

  $(x + y) \cdot (\overline{x} + z) \cdot (y + z) = (x + y) \cdot (\overline{x} + z)$
- $\overline{x + y} = \overline{x} \cdot \overline{y} \quad \overline{x \cdot y} = \overline{x} + \overline{y}$ **DeMorgan's Laws**

Proof of Minimization

$$\mathbf{x \cdot y + \bar{x} \cdot y = y} \qquad \mathbf{(x + y)(\bar{x} + y) = y}$$

x . y + x’ . y

= (x + x’) . y X(Y + Z) = XY + XZ (Distributive Law)

= 1 . y X + X’ = 1

= y X . 1 = X

Proof of DeMorgan's Laws

$$\overline{x + y} = \overline{x} \cdot \overline{y} \qquad \overline{x \cdot y} = \overline{x} + \overline{y}$$

Hints: It is important that we do not USE DeMorgan's Laws in doing this proof. This requires a different proof method. We will show that, x' . y', satisfies the definition of the complement of (x + y), defined as (x + y)' by DeMorgan's Law.

To show this we need to show that A + A' = 1 and A.A' = 0 with A = x + y and A' = x'. y'. This proves that x'. y' = (x + y)'.

Boolean Function Evaluation

$F1 = xy\overline{z}$
$F2 = x + \overline{y}z$
$F3 = \overline{x}\,\overline{y}\,\overline{z} + \overline{x}\,y\,z + x\overline{y}$
$F4 = x\overline{y} + \overline{x}\,z$

x	y	z	F1	F2	F3	F4
0	0	0	0	0		
0	0	1	0	1		
0	1	0	0	0		
0	1	1	0	0		
1	0	0	0	1		
1	0	1	0	1		
1	1	0	1	1		
1	1	1	0	1		

Expression Simplification

- **An application of Boolean algebra**
- **Simplify to contain the smallest number of literals (complemented and uncomplemented variables):**

$$A B + \overline{A} C D + \overline{A} B D + \overline{A} C \overline{D} + A B C D$$

$$= AB + ABCD + \overline{A} C D + \overline{A} C \overline{D} + \overline{A} B D$$

$$= AB + AB(CD) + \overline{A} C (D + \overline{D}) + \overline{A} B D$$

$$= AB + \overline{A} C + \overline{A} B D = B(A + \overline{A}D) + \overline{A}C$$

$$= B (A + D) + \overline{A} C \quad \text{5 literals}$$

Complementing Functions

- Use DeMorgan's Theorem to complement a function:
  1. Interchange AND and OR operators
  2. Complement each constant value and literal
- Example: Complement $F = \overline{x}y\overline{z} + x\overline{y}\overline{z}$

  $\overline{F} = (x + \overline{y} + z)(\overline{x} + y + z)$
- Example: Complement $G = (\overline{a} + bc)\overline{d} + e$

  $\overline{G} =$