CS 211: Computer Architecture
Digital Logic

Topics:

- Transistors
- Logic Gates
- Boolean algebra

Transistor: Building Block of Computers

Microprocessors contain millions (billions) of transistors

- Intel Pentium 4 (2000): 48 million
- IBM PowerPC 750FX (2002): 38 million
- IBM/Apple PowerPC G5 (2003): 58 million

Logically, each transistor acts as a switch

Combined to implement logic functions

- AND, OR, NOT

Combined to build higher-level structures

- Adder, multiplexer, decoder, register, …

Combined to build processor

CPU

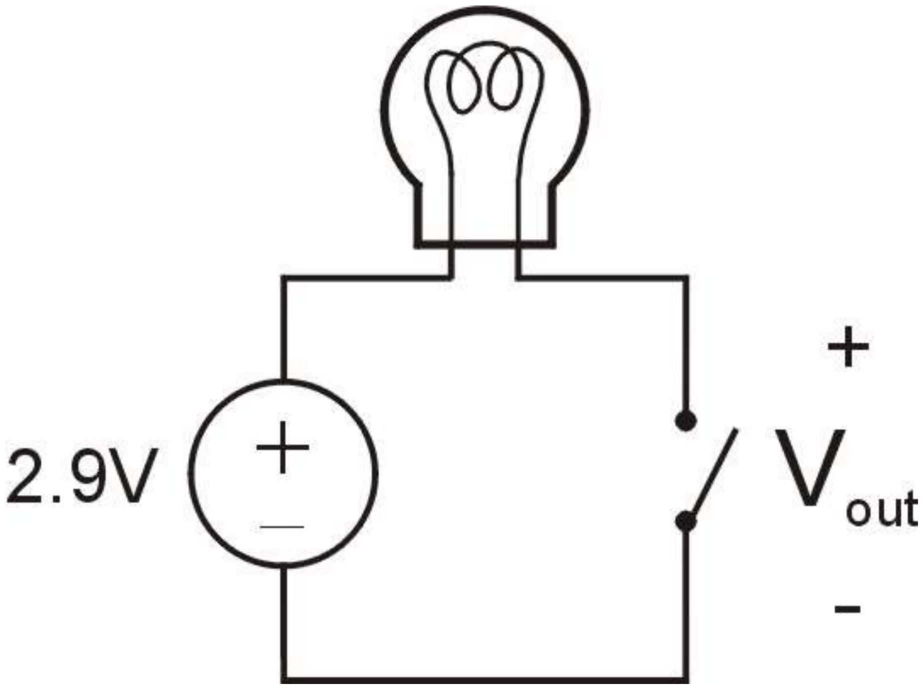
ALU

Logic gates

we will get to here — Transistors

silicon

Simple Switch Circuit



Switch open:

- No current through circuit
- Light is off
- V_{out} is +2.9V

Switch closed:

- Current flows
- Light is on
- V_{out} is 0V

Switch-based circuits **can easily represent two states: on/off, open/closed, voltage/no voltage.**

n-type MOS Transistor

normal

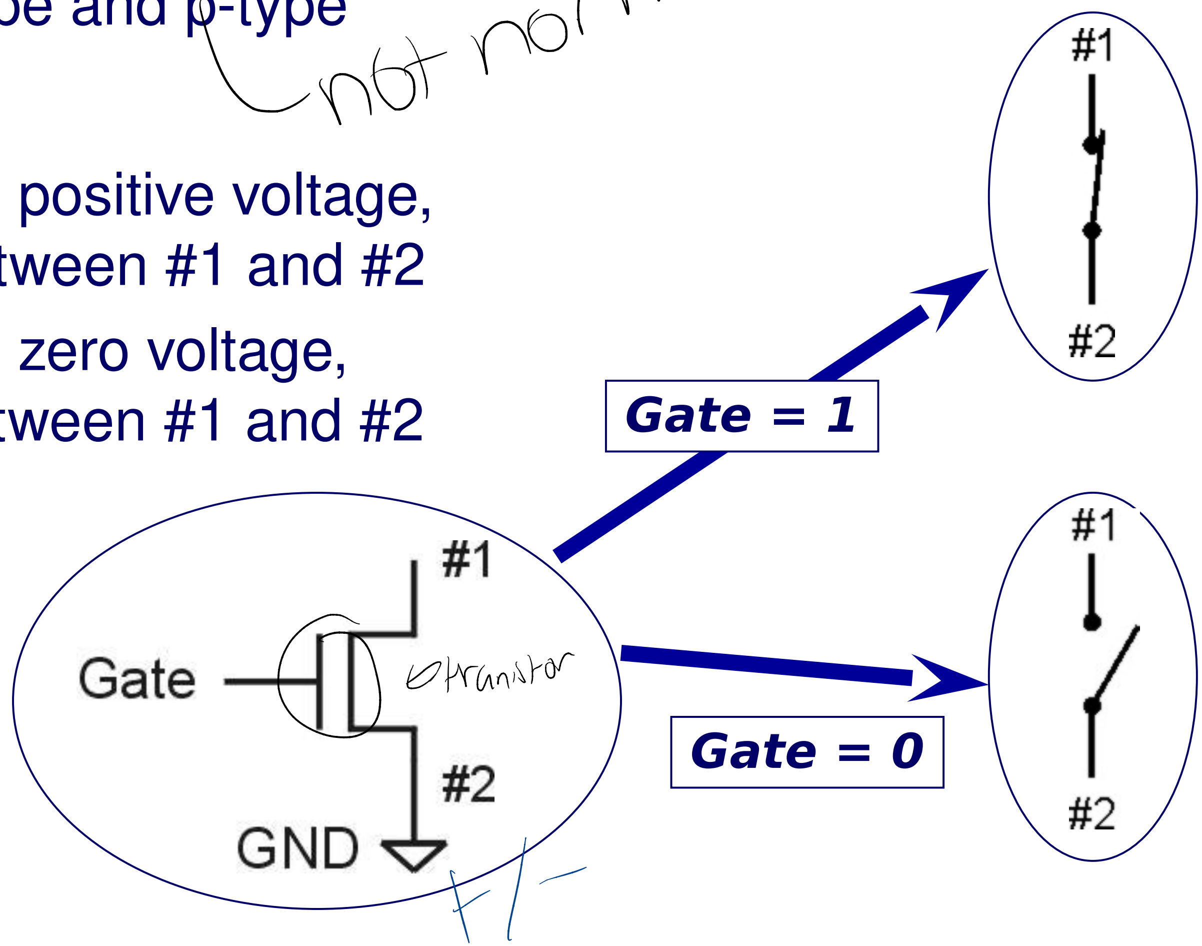
MOS = Metal Oxide Semiconductor

- two types: n-type and p-type

not normal

n-type

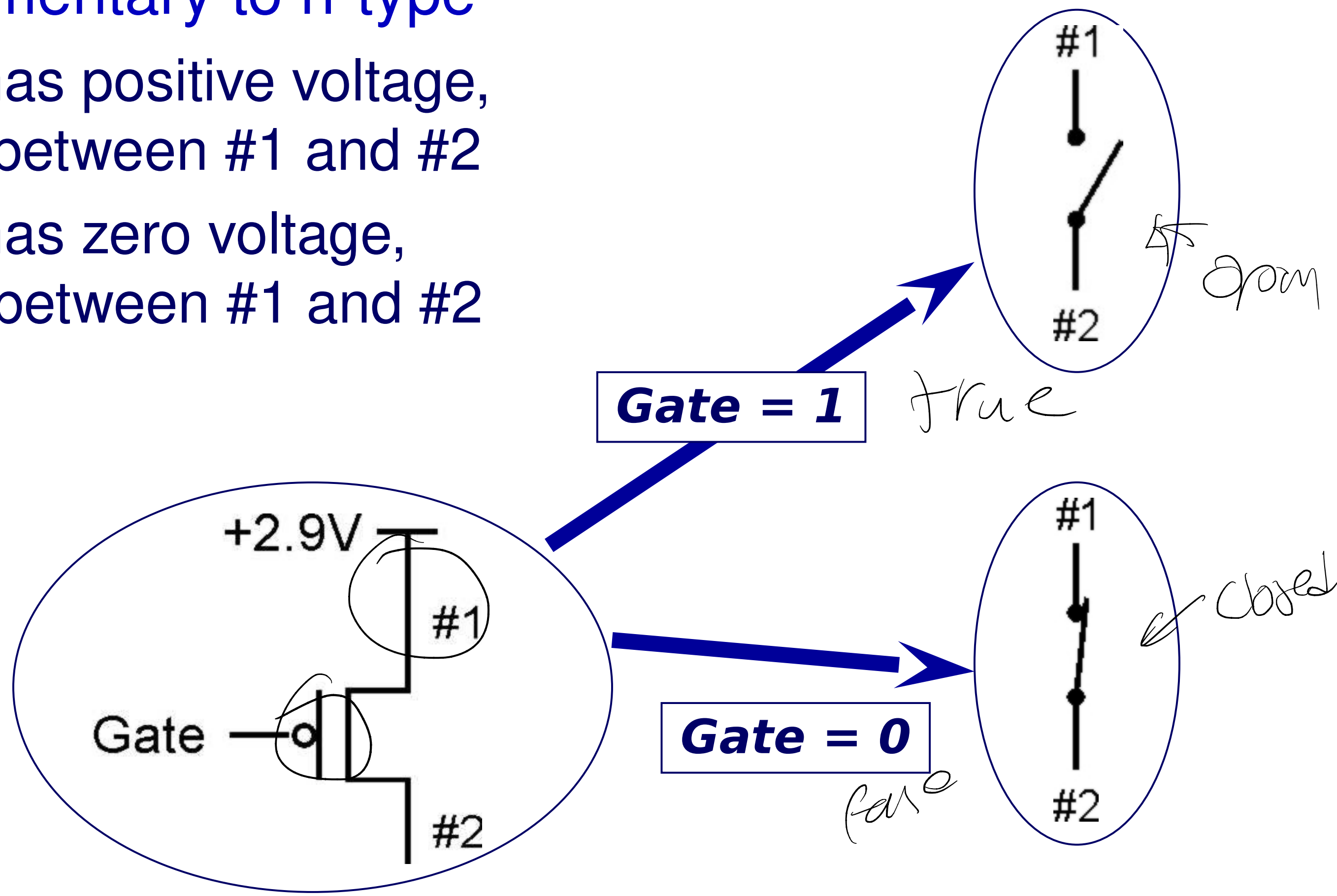
- when Gate has positive voltage, short circuit between #1 and #2
- when Gate has zero voltage, open circuit between #1 and #2



p-type MOS Transistor

p-type is complementary to n-type

- when Gate has positive voltage, open circuit between #1 and #2
- when Gate has zero voltage, short circuit between #1 and #2

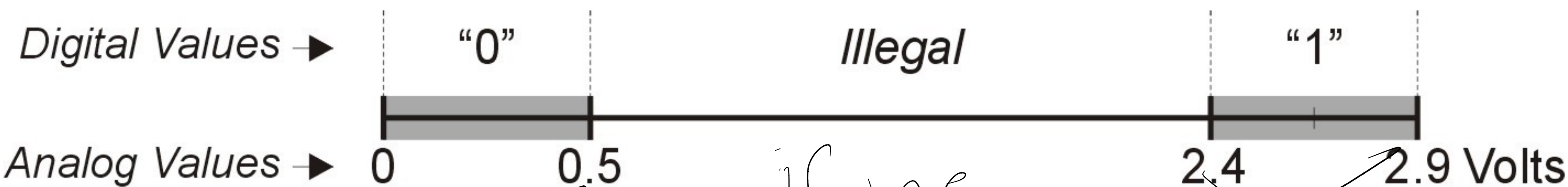


Logic Gates

Use transistors to implement logical functions: AND, OR, NOT

Digital symbols:

- recall that we assign a range of analog voltages to each digital (logic) symbol



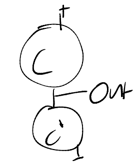
low enough

if we are not sure

high enough

- assignment of voltage ranges depends on electrical properties of transistors being used
  - typical values for "1": +5V, +3.3V, +2.9V
  - from now on we'll use +2.9V

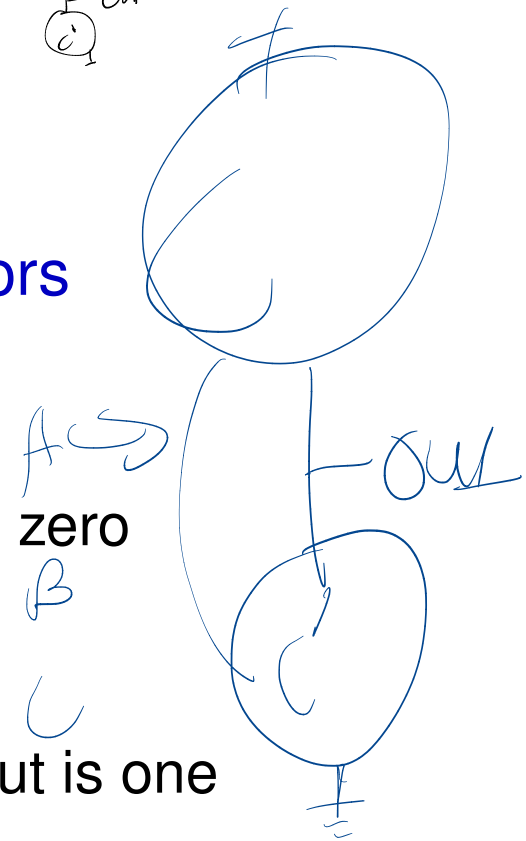
CMOS Circuit



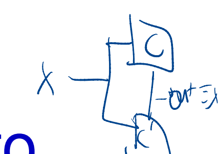
Complementary MOS

Uses both n-type and p-type MOS transistors

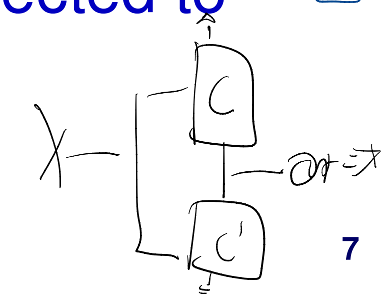
- p-type
  - Attached to + voltage
  - Pulls output voltage UP when input is zero
- n-type
  - Attached to GND
  - Pulls output voltage DOWN when input is one



MOS transistors are combined to form Logic Gates

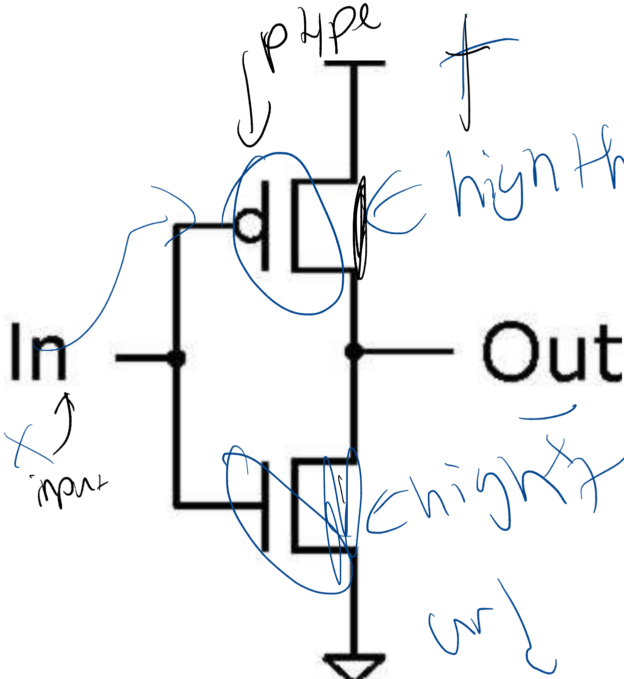


For all inputs, make sure that output is either connected to GND or to +, but not both!



Inverter (NOT Gate)

Easier to compute negation



p type

high than no connection

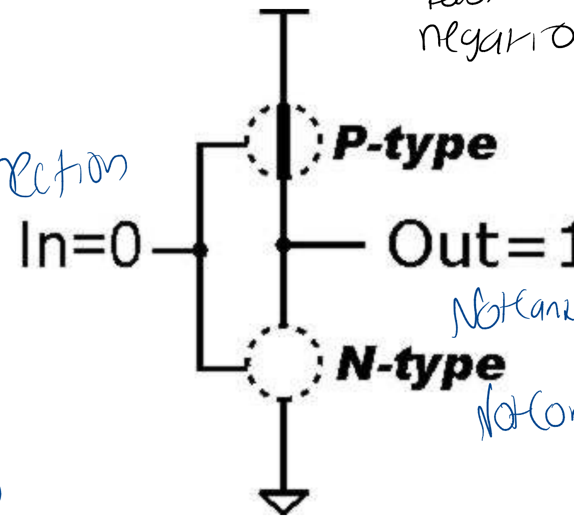
output $\bar{X}$

0

x input

high than a connection w/

ground



AND

OR

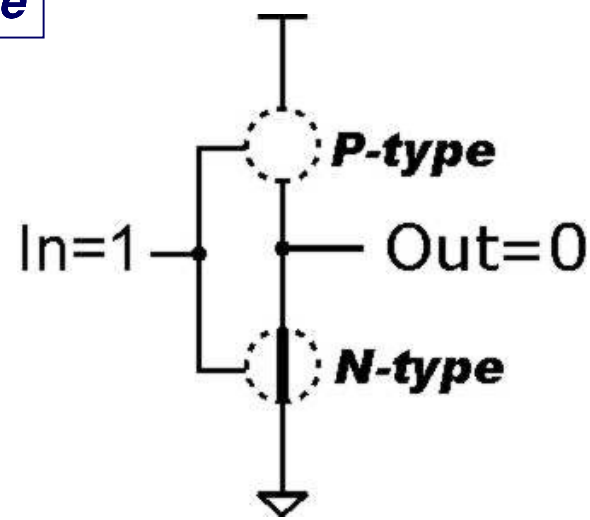
Not(and) = NAND

Not(or) = NOR

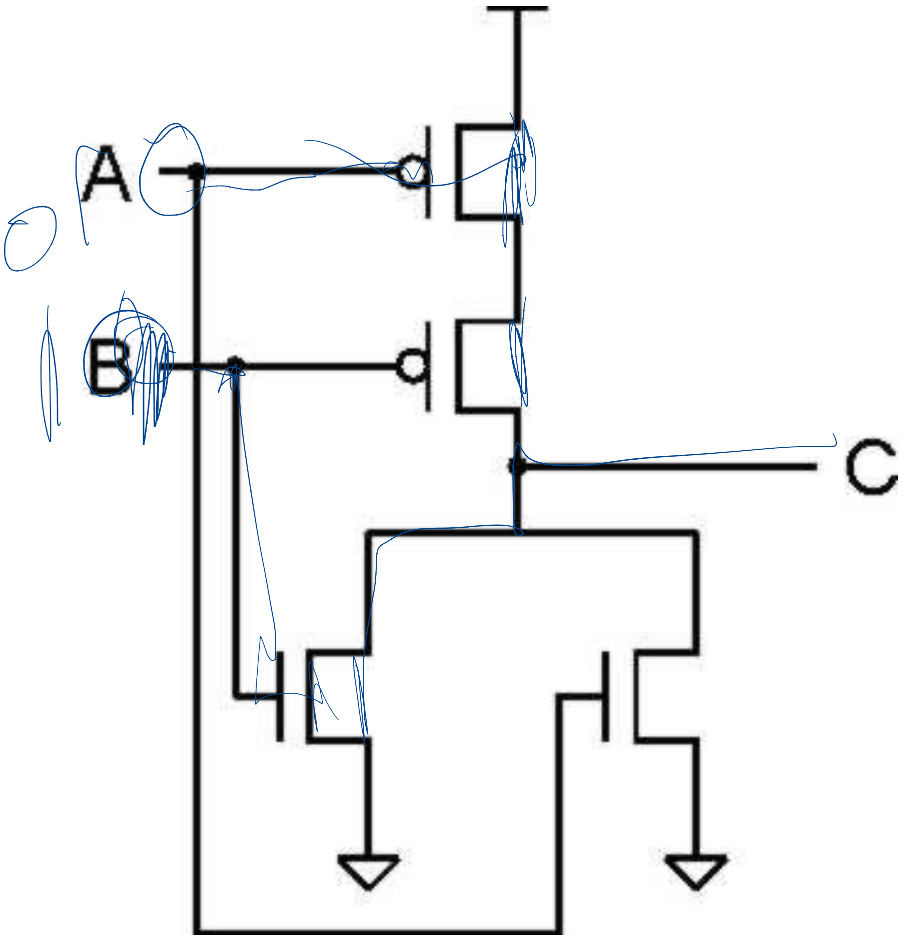
Truth table

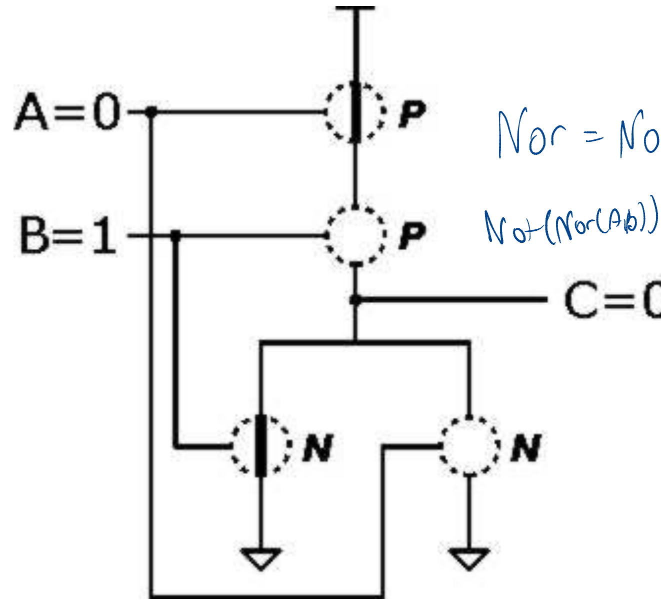
In	Out
0 V	2.9 V
2.9 V	0 V

In	Out
0	1
1	0



NOR Gate





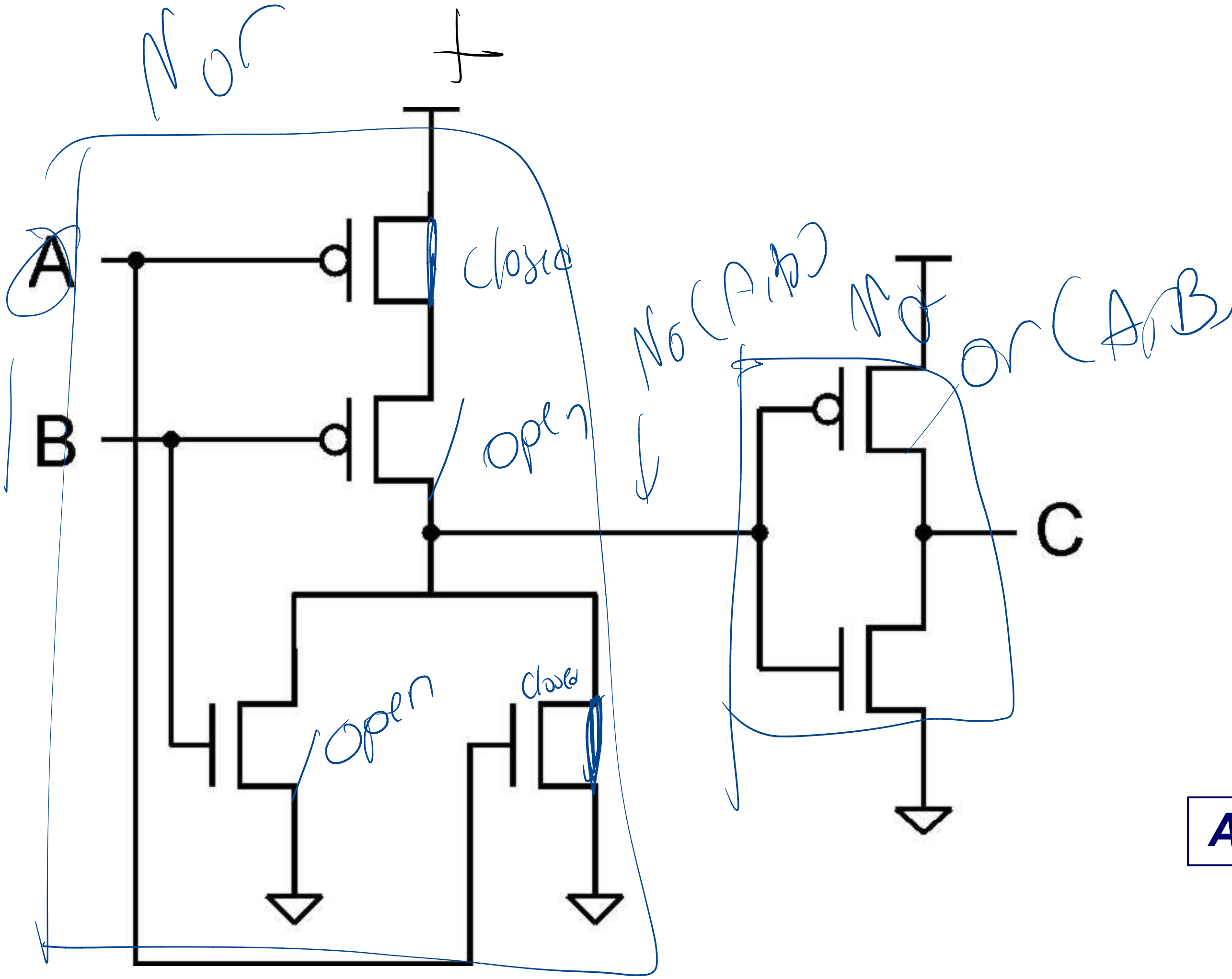
Nor = Not(or(A,B))

Not(Nor(A,b)) = Not(Not(or(A,B))) = Or(A,B)

A	B	C	A or B
0	0	1	0
0	1	0	1
1	0	0	1
1	1	0	1

Note: Serial structure on top, parallel on bottom.

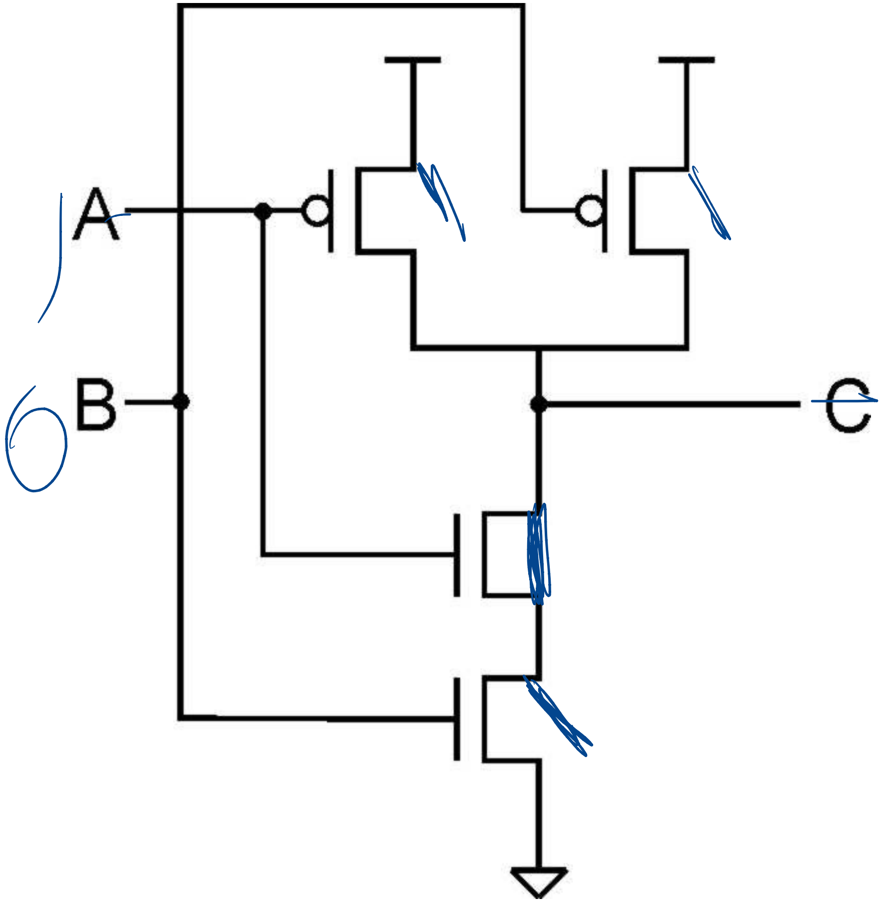
OR Gate

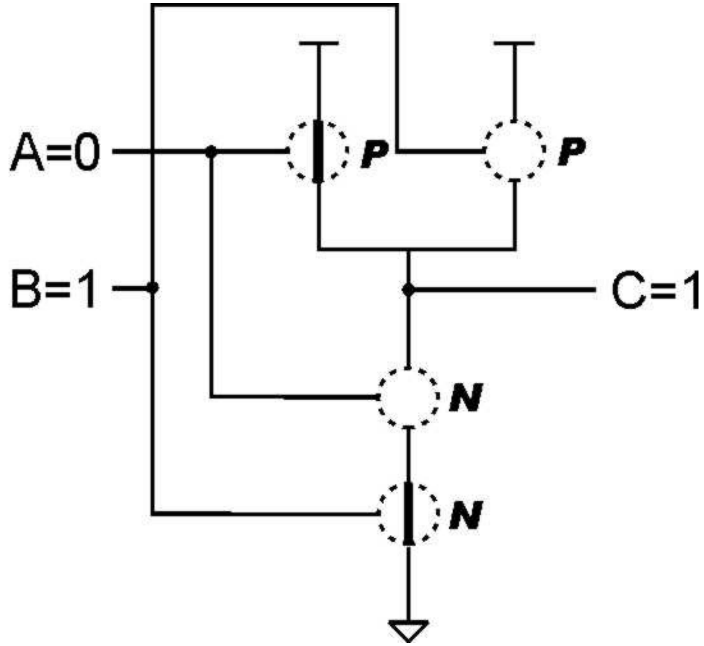


A	B	C
0	0	0
0	1	1
1	0	1
1	1	1

Add inverter to NOR.

NAND Gate (AND-NOT)

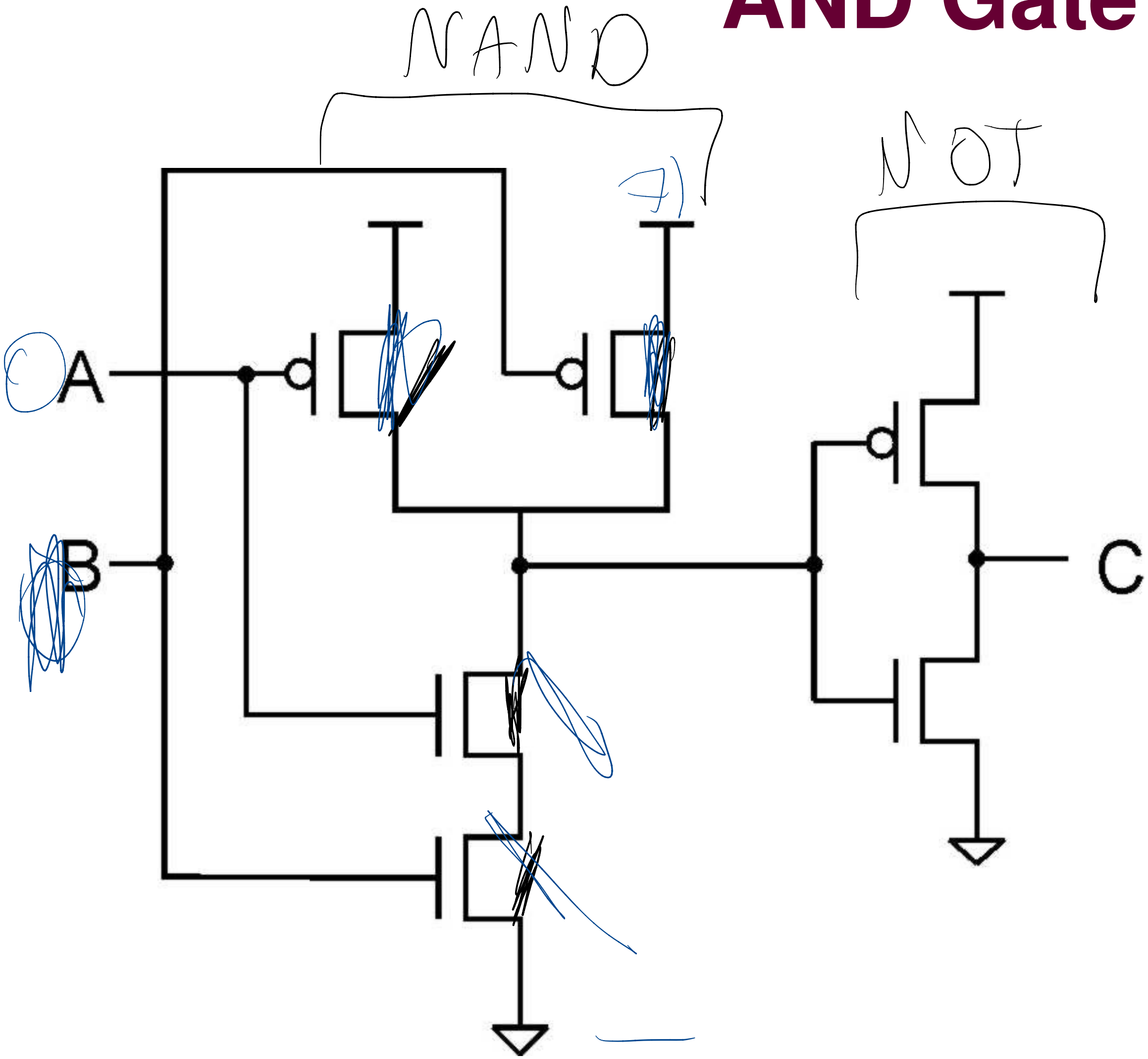




A	B	C
0	0	1
0	1	1
1	0	1
1	1	0

Note: Parallel structure on top, serial on bottom.

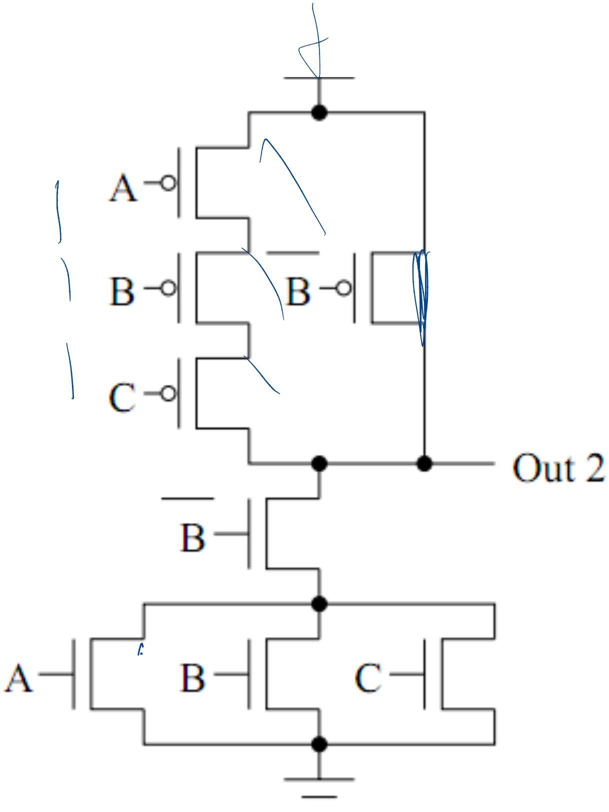
AND Gate



A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

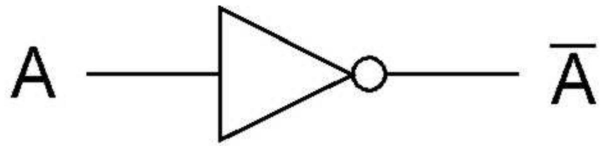
Add inverter to NAND.

What is this?

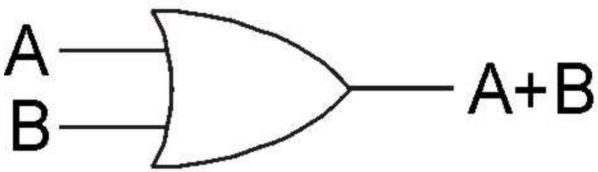


A	B	C	Out
0	0	0	**1**
0	0	1	**0**
0	1	0	**1**
0	1	1	**1**
1	0	0	**0**
1	0	1	**0**
1	1	0	**1**
1	1	1	**1**

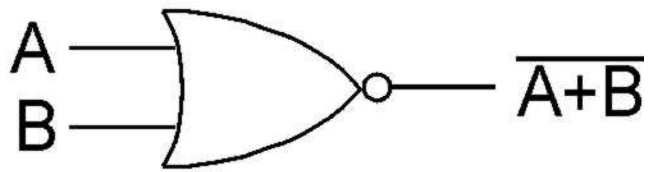
Basic Logic Gates Symbols



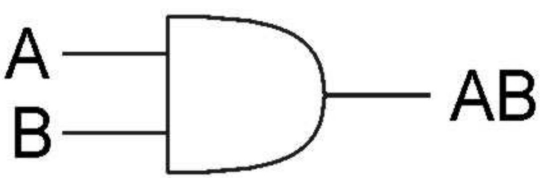
NOT



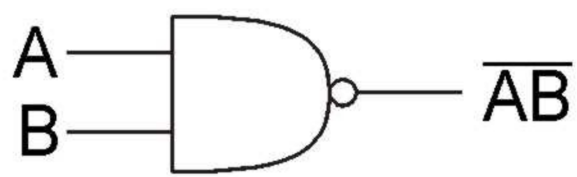
OR



NOR



AND



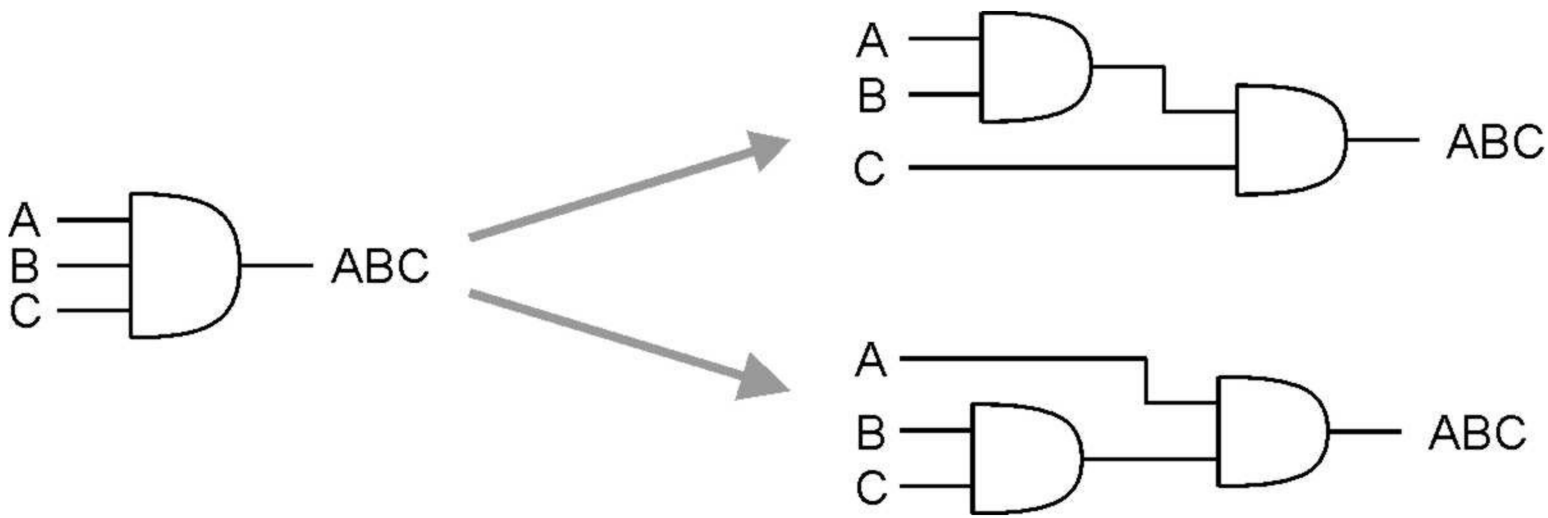
NAND

More than 2 Inputs?

AND/OR can take any number of inputs.

- AND = 1 if all inputs are 1.
- OR = 1 if any input is 1.
- Similar for NAND/NOR.

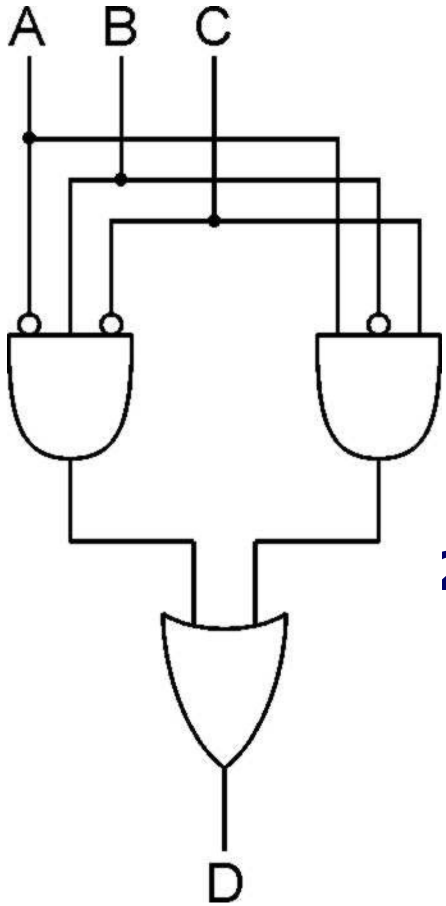
Can implement with multiple two-input gates.



Logical Completeness

Can implement ANY truth table with AND, OR, NOT.

A	B	C	D
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0



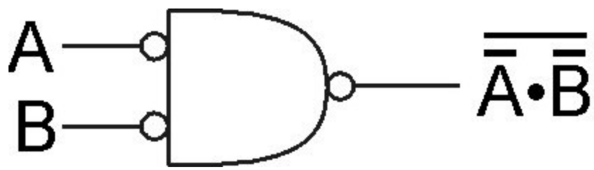
. AND combinations that yield a "1" in the truth table.

2. OR the results of the AND gates.

DeMorgan's Law

Converting AND to OR (with some help from NOT)

Consider the following gate:



To convert AND to OR (or vice versa), invert inputs and output.

A	**B**	$\overline{A}$	$\overline{B}$	$\overline{A}\cdot\overline{B}$	$\overline{\overline{A}\cdot\overline{B}}$
0	**0**	**1**	**1**	**1**	**0**
0	**1**	**1**	**0**	**0**	**1**
1	**0**	**0**	**1**	**0**	**1**
1	**1**	**0**	**0**	**0**	**1**

Same as A+B!

Generally, DeMorgan's Laws:

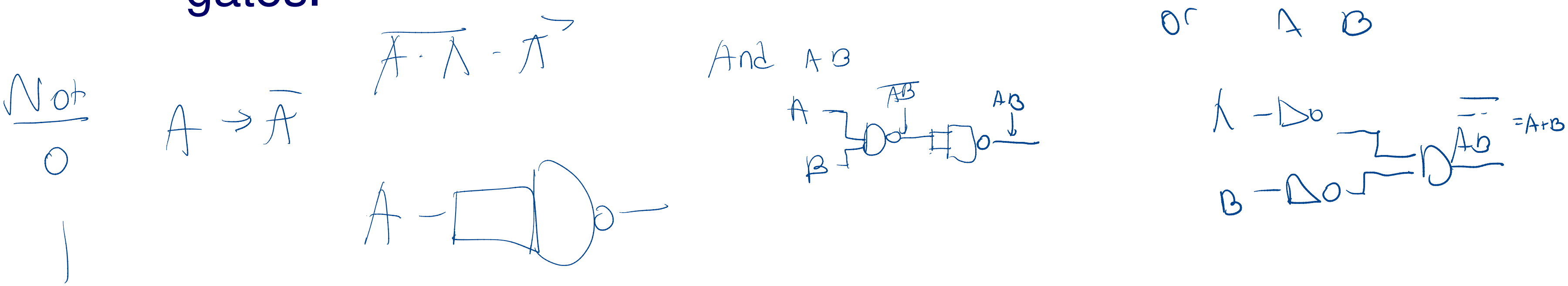
1. $\overline{PQ} = \overline{P} + \overline{Q}$
2. $\overline{P + Q} = \overline{P}\;\overline{Q}$

NAND and NOR Functional Completeness

Any gate can be implemented using either NOR or NAND gates.

Why is this important?

- When building a chip, easier to build one with all of the same gates.



NAND, NOR universality

T F F F F

$\overline{(\bar{a}\,\overline{ab})(\overline{b}\,\overline{ab})}$

$(\bar{a}+ab)(\bar{b}+\bar{a}+\bar{b})$

NAND, NOR universal because they can realize AND, OR, NOT

$(AB+\bar{A})(AB+B)\ \overline{\overline{AB}\,A} + \overline{\overline{AB}\,B}$

$(\overline{AB}+\bar{A})(\overline{AB}+B)\ \overline{\overline{AB}\,A} + \overline{\overline{AB}\,B}$

$\bar{A}$ = A NAND A	$\bar{A}$ = A NOR A
AB = $\overline{\text{A NAND B}}$	A+B = $\overline{\text{A NOR B}}$
A+B = $\bar{A}$ NAND $\bar{B}$	AB = $\bar{A}$ NOR $\bar{B}$

Terminology (Boolean Algebra)

Binary variable: a symbolic representation that might be 0 or 1 (eg. X, Y, A, B)

Complement: the opposite value of variable X

Literal: a boolean variable or its complement (eg, X, $\overline{X}$)

Expression: a set of literals combined with logical operations (eg. AB + C)

Boolean algebra

- Values: 0, 1
- Operations: and, or, not, xor, implies, etc.
  - X AND Y: like multiplication
  - X XOR Y: like addition (mod 2)

A + B

	0	1
0	0	0
1	0	1

	0	1
0	0	1
1	1	0

Boolean Identities

$X + YZ + YX + YZ$

OR	AND	NOT	
$X+0 = X$	$X1 = X$		(identity)
$X+1 = 1$	$X0 = 0$		(null)
$X+X = X$	$XX = X$		(idempotent)
$X+\overline{X} = 1$	$X\overline{X} = 0$		(complementarity)
		$\overline{\overline{X}} = X$	(involution)
$X+Y = Y+X$	$XY = YX$		(commutativity)
$X+(Y+Z) = (X+Y)+Z$	$X(YZ) = (XY)Z$		(associativity)
$X(Y+Z) = XY + XZ$	$X+YZ = (X+Y)(X+Z)$		(distributive)
$\overline{X+Y} = \overline{X}\,\overline{Y}$	$\overline{XY} = \overline{X} + \overline{Y}$		(DeMorgan's theorem)

$X + XZ + XY + YZ$

Boolean Algebra Example

$$F = \overline{X}YZ + \overline{X}Y\overline{Z} + XZ$$

$\overline{X}Y(Z + \overline{Z}) + XZ$ (by reverse distribution)

$\overline{X}Y1 + XZ$ (by complementarity)

$\overline{X}Y + XZ$ (by identity)

Boolean Algebra Example 2

Find the complement of F

$(A\bar{B})(\bar{A}B)$

$(\bar{A}+\bar{\bar{B}}) \cdot (\bar{\bar{A}}+B)$

$(\bar{A}B)(A\bar{B})$

$(\bar{A}+B)(A+\bar{B})$

$$F = A\bar{B} + \bar{A}B$$

$$\bar{F} = \overline{A\bar{B} + \bar{A}B}$$

$$\overline{(A\bar{B})}\;\overline{(\bar{A}B)} \qquad \text{(by DeMorgan's)}$$

$$(\bar{A} + \bar{\bar{B}})(\bar{\bar{A}} + \bar{B}) \qquad \text{(by DeMorgan's)}$$

$$(\bar{A} + B)(A + \bar{B}) \qquad \text{(by involution)}$$

Using DeMorgan's Laws to Complement

1. Big bar over AND and OR of 2 or more functions
2. Replace AND with OR, OR with AND
3. 1 with 0, 0 with 1
4. F with not(F), not(F) with F

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

$(A\overline{B}C) + (ACD) + (B+\overline{C})$

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

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

$A\overline{B}CD$
$+ A\overline{B}CD$
$A\overline{B}CD$

$F = \overline{A\overline{B}C},\ G = \overline{ACD},\ H = B\overline{C},\ \overline{F+G+H} = \overline{F}\,\overline{G}\,\overline{H}$

$$= (A\overline{B}C)(ACD)(\overline{B\overline{C}})$$

$(A\overline{B}C)(ACD) = A\overline{B}CD,\ F = B,\ G = \overline{C},\ \overline{FG} = \overline{F}+\overline{G}$

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

$$= A\overline{B}CD + A\overline{B}CD$$

$$= A\overline{B}CD$$

Duals

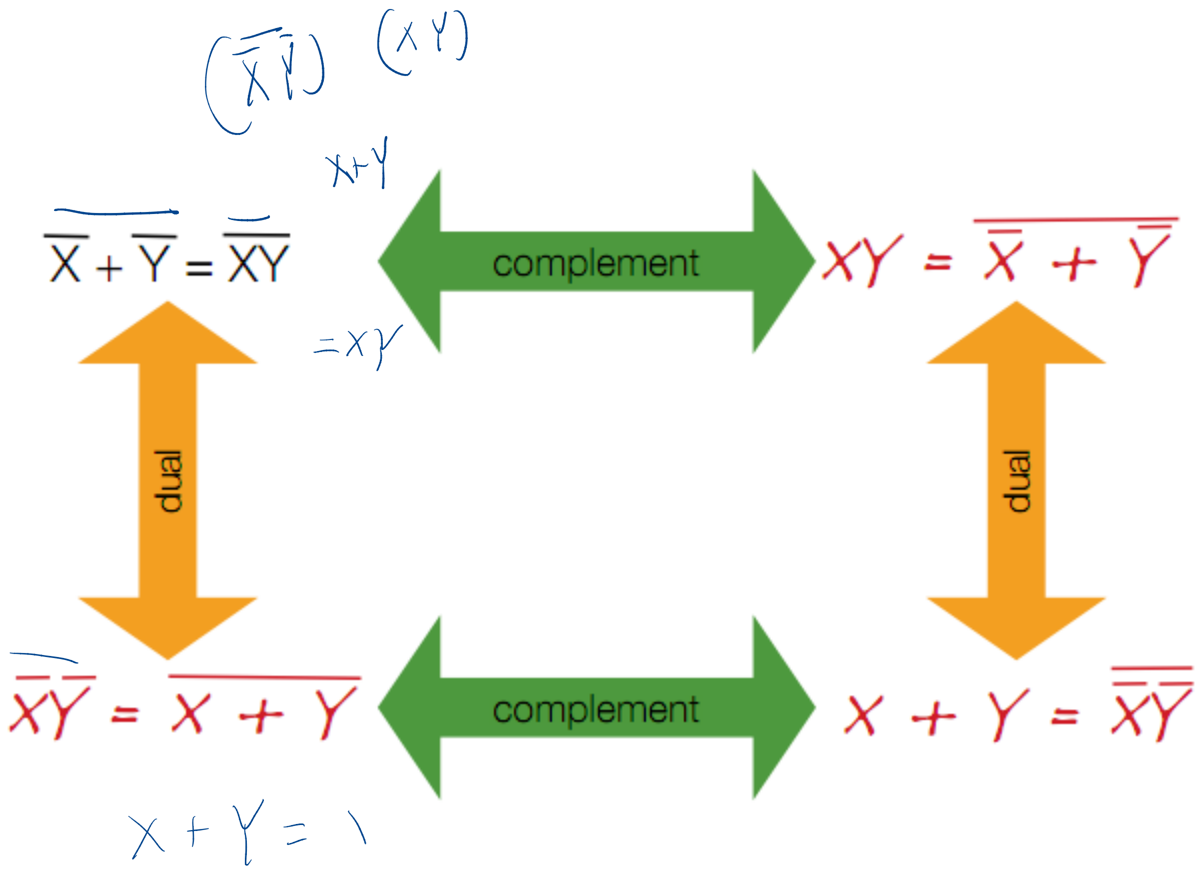
All boolean expressions have duals

Any theorem you prove, you can also prove for the dual

To form a dual

1. replace AND with OR, OR with AND
2. replace 1 with 0, 0 with 1

Complements and Duals



Complement Using Duals

$A(Z + X(YW))$

Get dual and then complement each literal

$(\bar{X})(A + ZX + YW)(Y + ZW)$

$$F = \bar{X} + A\,(Z + \bar{X}\,(Y + W) + \bar{Y}\,(Z + W))$$

$(X)(\bar{A} + \bar{Z}(X + \bar{Y}\bar{W})(Y + \bar{Z}\bar{W}))$

$$\text{Dual: } F_{dual} = X\,(A + Z\,(\bar{X} + YW)(\bar{Y} + ZW))$$

$$\bar{F} = \bar{X}\,(\bar{A} + \bar{Z}\,(X + \bar{Y}\bar{W})(Y + \bar{Z}\bar{W}))$$

Simplifying Expressions

$\bar{\bar{A}}BC + A\bar{B}C + AB(\bar{C}+C)$

$\bar{A}BC + A\bar{B}C + AB$

$\bar{A}BC + A(\bar{B}C + B)$

$$\bar{A}BC + A\bar{B}C + AB\bar{C} + ABC$$

$\bar{A}BC + A(B+C)$

$\bar{A}BC + AB + AC$

$B(\bar{A}C + A) + AC$

$B(A+C) + AC$

$AB + BC + AC$

$$\downarrow$$

$$BC(\bar{A} + A) + A\bar{B}C + AB\bar{C}$$

$$\downarrow$$

$$BC(1) + A\bar{B}C + AB\bar{C}$$

$$\downarrow$$

$$BC + A\bar{B}C + AB\bar{C}$$

$$\downarrow$$

$$B(C + A\bar{C}) + A\bar{B}C$$

$(C+A)(C+\bar{C})$

$C + AC + A\bar{C}$

$C + A(C+\bar{C})$

$C + A(1) \Rightarrow C + A$

$$B(C + A) + A\bar{B}C$$

$$\downarrow$$

$$BC + AB + A\bar{B}C$$

$$\downarrow$$

$$BC + A(B + \bar{B}C)$$

$(B+\bar{B})(B+C)$

$B + BC + \bar{B}C$

$B + C(B+\bar{B})$

$B + C$

$$\downarrow$$

$$BC + A(B + C)$$

$$\downarrow$$

$$BC + AB + AC$$