RUTGERS

THE STATE UNIVERSITY
OF NEW JERSEY

Computer Architecture (CS-211)
Recitation-8

Song Wen

Topics

- Transistors
- Logic Gates
- Boolean Algebra

* Some materials are collected and compiled from previous year's CS 211 lectures and TAs

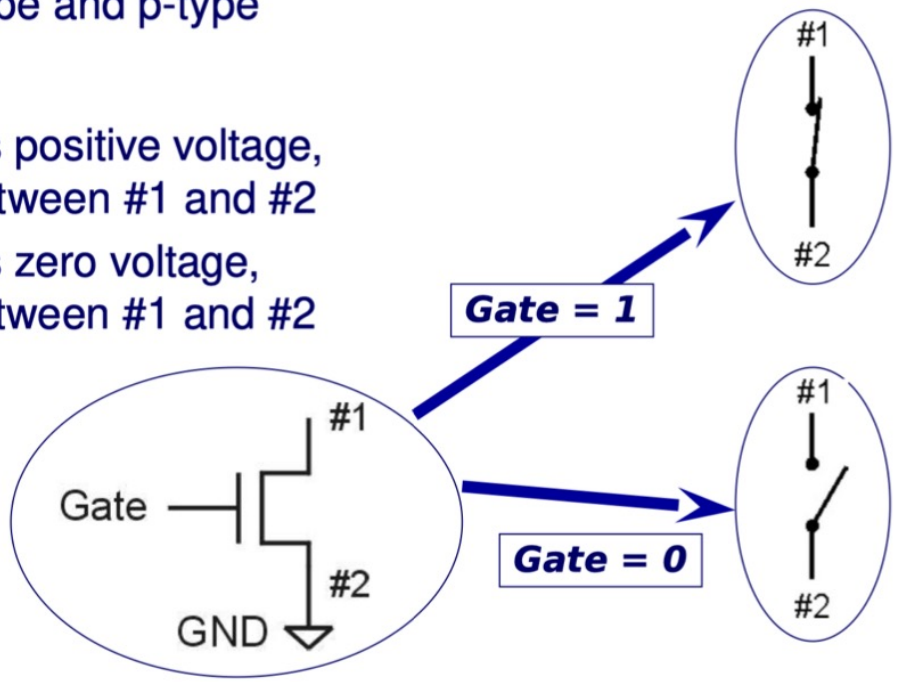
n-Type MOS Transistor

MOS = Metal Oxide Semiconductor

- two types: n-type and p-type

n-type

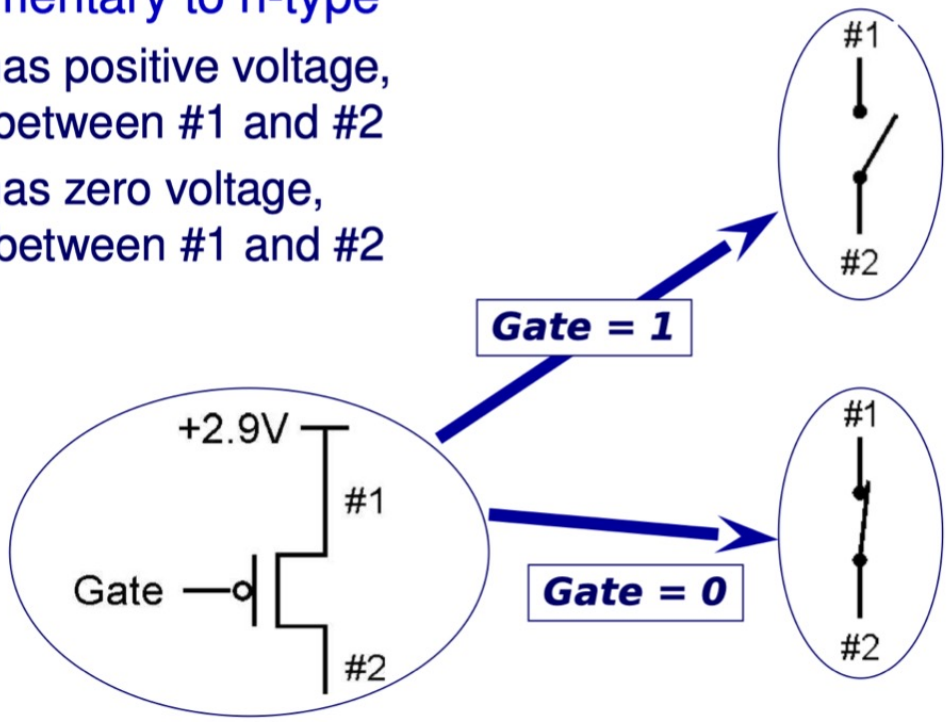
- when Gate has positive voltage, short circuit between #1 and #2
- when Gate has zero voltage, open circuit between #1 and #2



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

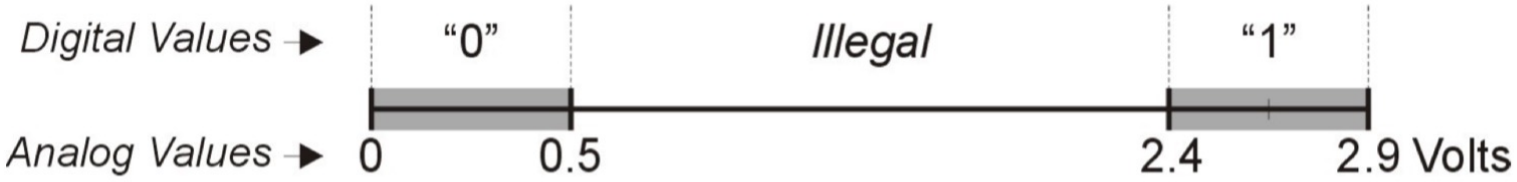


Logic Gate

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

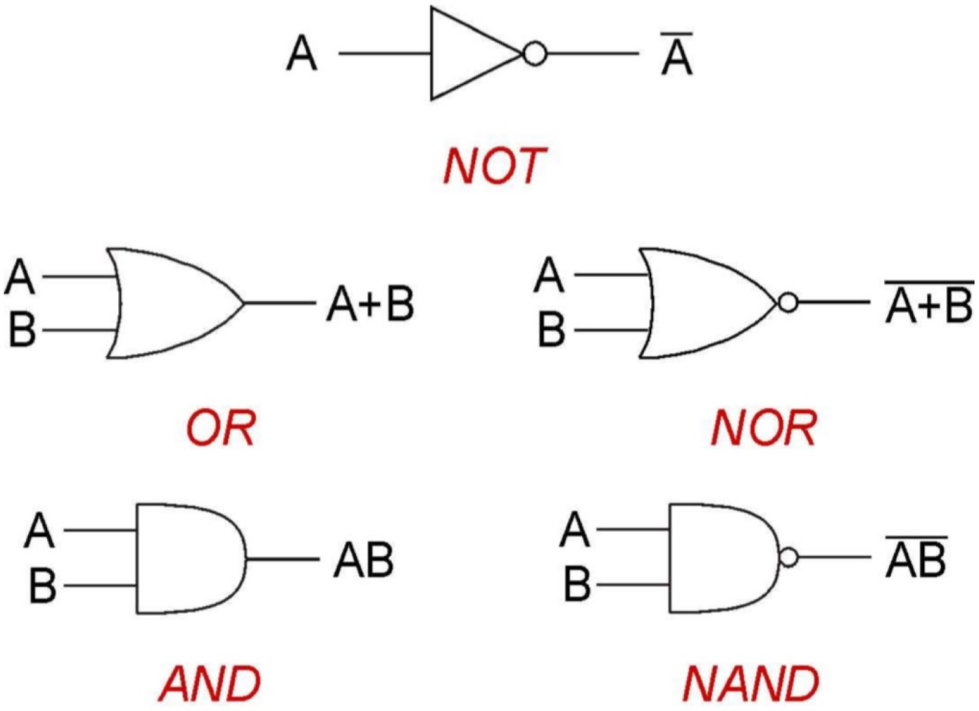
Digital symbols:

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



- 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

Basic Logic Gates Symbols

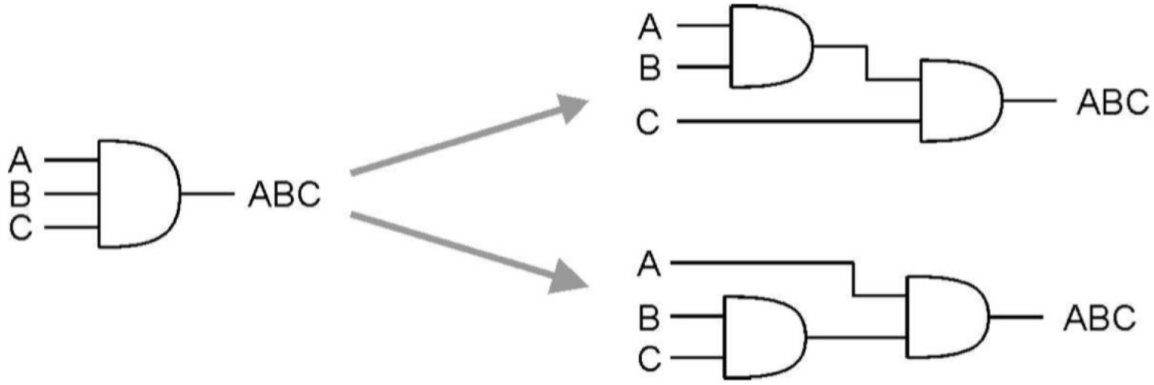


More than 2 inputs?

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.

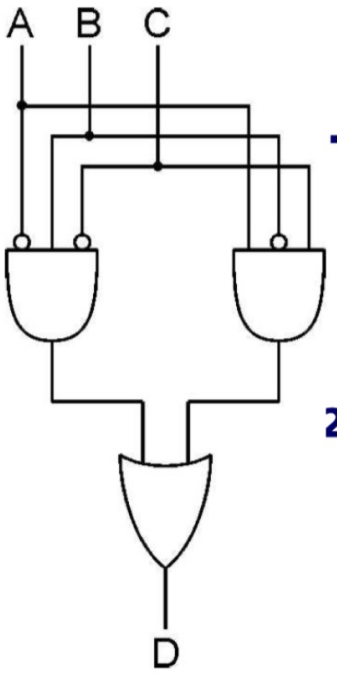
Can implement with multiple two-input gates.



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



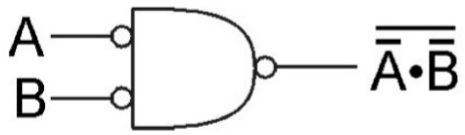
. AND combinations that yield a "1" in the truth table.

2. OR the results of the AND gates.

DeMorgan's Law

Converting AND to OR (with some help from NOT)

Consider the following gate:



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!

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

Generally, DeMorgan's Laws:

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

NAND, NOR universality

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

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

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.

Boolean Identities

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)

Q&A

Thanks!