Georgia Tech College of Engineering
School of Electrical
and Computer Engineering

ECE 2020 Lecture 3:
Switching and Transistors

Instructor: Samuel Talkington

September 3, 2024

Logistics

- Extra office hours happened today! Thank you for the excellent questions. Any additional questions?
- **Thursday office hours:** Normal schedule, 11:30am-1:00pm.
- **Adjusted Friday office hours** this week: 12:30pm-2:00pm in person.
- First homework assignment due September 4th, 2024, 11:59pm.
- Updated course schedule coming.

Agenda

- Recap of digital signals and boolean algebra
- Switching logic signals and truth tables
- Transistors
- Manipulating digital signals with Boolean logic
- Understanding switching operations

Recap

Boolean variables

A **Boolean variable**, also known as a **binary variable**, is a variable that can take the values **one** or **zero** depending on whether some other thing is **true** or **false**, i.e.:

$$X = \begin{cases} 1 & \text{if some thing is true} \\ 0 & \text{if some thing is false.} \end{cases} \tag{1}$$

Digital signal

A digital signal, also known as a **binary function**, **Boolean function**, or **logic function**, is a function F that receives **a collection of binary/boolean variables** $X, Y, \dots$ as input:

$$Z = F(X, Y, \dots), \tag{2}$$

then performs a sequence of **logical operations**, and returns another binary Z.

Axioms of Boolean Algebra

Boolean Axioms

The fundamental axioms of Boolean Algebra are the following:

(1) $X \neq 0 \implies X = 1$, conversely $X \neq 1 \implies X = 0$

(2) $X = 0 \implies \overline{X} = 1$, conversely $X = 1 \implies \overline{X} = 0$

(3) $0 \cdot 0 = 0$, similarly $1 + 1 = 1$

(4) $1 \cdot 1 = 1$, similarly $0 + 0 = 0$

(5) $0 \cdot 1 = 1 \cdot 0 = 0$, similarly $0 + 1 = 1 + 0 = 1$

Everything else can be proven from these Axioms!

Logic gates: Making logic real

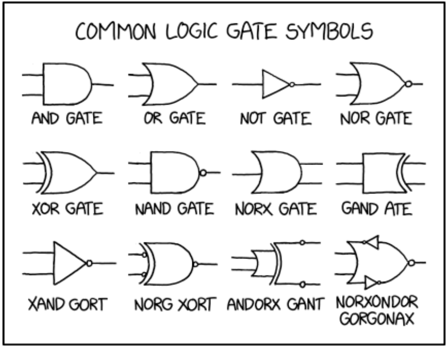
Logic gates

Are the fundamental building blocks of digital circuits.

The key idea

Gates = Operators

Circuits = Functions



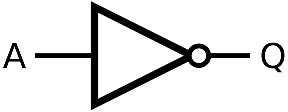
Source: xkcd

The NOT operator

Truth table

A	$Q = \overline{A}$
0	1
1	0

Logic gate



The NOT gate

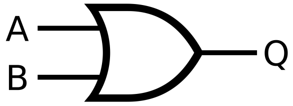
Note: a bubble can be added to other gates to complement the input(s)/output(s).

The OR operator

Truth table

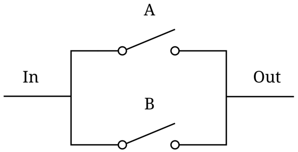
A	B	$Q = A + B$
0	0	0
0	1	1
1	0	1
1	1	1

Logic gate



The OR gate

Switch logic

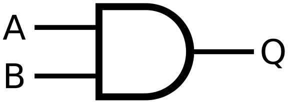


The AND operator

Truth table

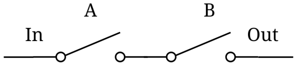
A	B	$Q = A \cdot B$
0	0	0
0	1	0
1	0	0
1	1	1

Logic gate



The AND gate

Switch logic

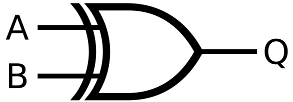


The XOR operator

Truth table

A	B	$Q = A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

Logic gate



The XOR gate

DeMorgan's Theorem

Assumption: Suppose that you are given *any* two binary signals

$$X, Y = \begin{cases} 1 & \text{if something is true} \\ 0 & \text{if something is false} \end{cases}$$

Then, the following two logical relationships are *always* true:

$$\overline{X + Y} = \overline{X} \cdot \overline{Y} \tag{3a}$$

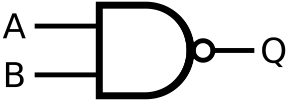
$$\overline{X \cdot Y} = \overline{X} + \overline{Y} \tag{3b}$$

The NAND operator

Truth table

A	B	$Q = \overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

Logic gate



The NAND gate

Connection

Implements DeMorgan's Theorem:

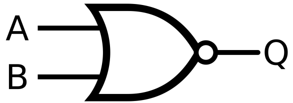
$$\overline{A \cdot B} = \overline{A} + \overline{B}$$

The NOR operator

Truth table

A	B	$Q = \overline{A + B}$
0	0	1
0	1	0
1	0	0
1	1	0

Logic gate



The NOR gate

Connection

Implements DeMorgan's Theorem:

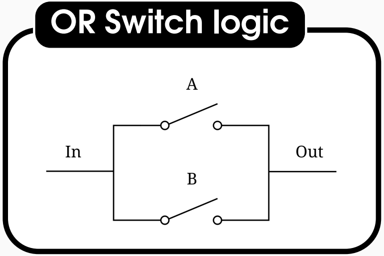
$$\overline{A + B} = \overline{A} \cdot \overline{B}$$

Switching

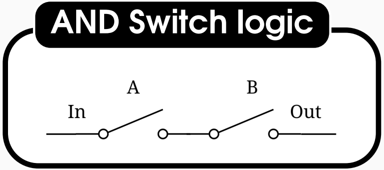
A very simple, but powerful idea

A **switch** is a simple but powerful idea: **Either electricity flows through a circuit or it does not So,** we can represent logic via switches too!

Parallel switches: Implementation of “addition”: logical OR



Series switches: Implementation of “multiplication”: logical AND



Example: Simplification, drawing gate and switch logic

Example

Consider the following logic function:

$$F(A,B,C) = \overline{\left(A \cdot B + \overline{A} \cdot B \cdot \overline{C} + A \cdot C\right)} \cdot \left(A + \overline{B \cdot C}\right) \tag{4}$$

Simplify the expression as much as possible using Boolean Algebra theorems, draw the switching logic, and draw the gate-level schematic.

Solution: Simplification

$$F(A,B,C) = \overline{\left(A \cdot B + \overline{A} \cdot B \cdot \overline{C} + A \cdot C\right)} \cdot \left(A + \overline{B \cdot C}\right)$$

Solution: Switch Logic

Solution: Gate Logic

Transistors

Transistors

We're all sort of familiar with switches. But these days, there's a better device—a *transistor*

What is CMOS?

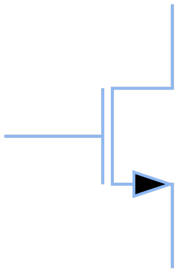
CMOS

A **complementary metal oxide semiconductor transistor** (CMOS) is a 3-terminal device that acts like a voltage-controlled resistance. It is a **powered switch** that can be controlled with a voltage.

In digital logic, a MOS transistor is operated so its resistance is always either very high (and the transistor is “off”) or very low (and the transistor is “on”).

NMOS

NMOS diagram

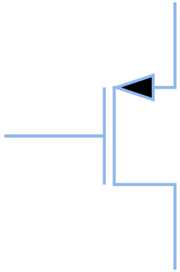


Logic

$$R_{ds} = \begin{cases} \text{large } \Omega & v_{gs} \leq 0 \\ \text{small } \Omega & v_{gs} > 0 \end{cases}$$

PMOS

PMOS diagram



Logic

$$R_{ds} = \begin{cases} \text{small } \Omega & v_{gs} < 0 \\ \text{large } \Omega & v_{gs} \geq 0 \end{cases}$$

Transistors

Transistors: They are just cool-looking switches.

We can use NMOS and PMOS transistors (switches) to design logical operations in real life!

Example: What is switching logic for a NOT gate?

Inverter design example

Example

Suppose you want to implement, with CMOS switches, the NOT gate (aka, inverter):

$$v_{\text{out}} = \overline{v_{\text{in}}}$$

How would this be done?

Inverter design example

CMOS diagram