

Lecture Outline:

- Laws of Logic
- Logical Equivalence
 - Truth-Tables to Boolean Formulae to Circuits
- Universality of NAND

1 Logic Statements

Logic statements can be viewed as ordinary English sentences that are either true or false. For instance, “Lions make good pets” is a Logic statement. So is “Giraffes are tall”. Logic is also intimately connected with electronic circuits where computation is being performed over binary numbers. We can see the following equivalence.

Binary (mathematical)	Mechanical	Electrical	Logic (mathematical)
1	on (pushed)	+5V	T
0	off (unpushed)	0V	F

2 Laws of Logic

We are all familiar with elementary laws of arithmetic:

Commutativity: $a + b = b + a$, $a \times b = b \times a$

Associativity: $(a + b) + c = a + (b + c) = a + b + c$ (same for multiplication)

Distributivity: $a \times (b + c) = a \times b + a \times c$

Identity: $a + 0 = a$; $a \times 1 = a$

Inverse: $a + (-a) = 0$; $a \times (1/a) = 1$;

We now present analogous laws in logic

Commutativity: $p \vee q = q \vee p$; $p \wedge q = q \wedge p$

Associativity: $p \vee (q \vee r) = (p \vee q) \vee r$, similarly for \wedge

Distributivity: $p \wedge (q \vee r) = (p \wedge q) \vee (p \wedge r)$; $p \vee (q \wedge r) = (p \vee q) \wedge (p \vee r)$

Identity: $p \vee F = p$; $p \wedge T = p$

Complement: $p \vee \neg p = T$; $p \wedge \neg p = F$

De Morgan's Law: $\neg(p \vee q) = \neg p \wedge \neg q$; $\neg(p \wedge q) = \neg p \vee \neg q$

Analogous laws also apply to sets, and can be nicely illustrated using Venn diagrams, as we will see later in the course.

3 Converting truth tables to Boolean formulae

The simplest approach is to use the disjunctive normal form (DNF). Identify all rows of the truth table in which the output is T. Express each of these rows as a conjunction. The required Boolean formula is a disjunction of these conjuncts. Here is a simple example.

a	b	out
0	0	1
0	1	1
1	0	0
1	1	1

By DNF construction, we obtain:

$$\begin{aligned} & (\neg a \wedge \neg b) \vee (\neg a \wedge b) \vee (a \wedge b) \\ = & (\neg a \wedge (\neg b \vee b)) \vee (a \wedge b) && \text{(distributivity)} \\ = & (\neg a \wedge T) \vee (a \wedge b) && \text{(complement)} \\ = & \neg a \vee (a \wedge b) && \text{(identity)} \\ = & (\neg a \vee a) \wedge (\neg a \vee b) && \text{(distributivity)} \\ = & T \wedge (\neg a \vee b) && \text{(complement)} \\ = & \neg a \vee b. \end{aligned}$$

One can also apply DNF construction via 0s.

$$\begin{aligned} & \neg(a \wedge \neg b) \\ = & \neg a \vee \neg \neg b && \text{(De Morgan)} \\ = & \neg a \vee b && \text{(double negation).} \end{aligned}$$

4 Converting Boolean formulae to circuits

This is straightforward. Replace each of the \wedge , \vee , and \neg by AND, OR, and NOT gates, respectively, feeding in the inputs, connecting the gates, and duplicating the wires as necessary.

For example for the formula above, we get the following circuit:

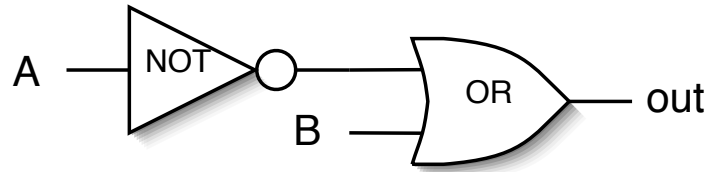


Figure 1: Circuit for Boolean formula $a \vee \neg b$

5 Converting circuits to truth tables

This is also straightforward. All one needs to do is evaluate the output for every possible combination of input values.

6 Universality of NAND

We now show that using NAND gates, we can build OR, AND, and NOT gates and, thus, all combinational circuits. Hence, NAND is referred to as a universal gate. The following figure depicts how the basic gates can be constructed entirely out of NAND gates.

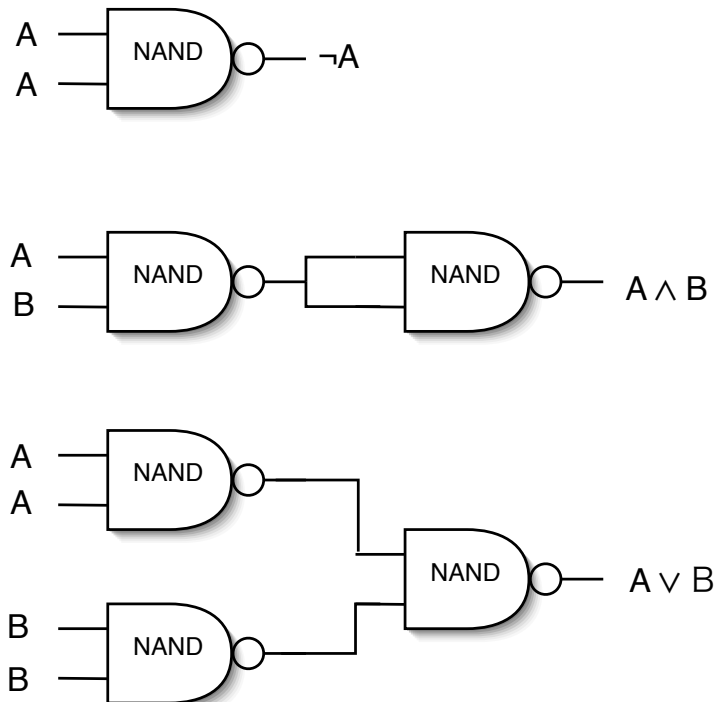


Figure 2: Universality of NAND