CS1101: Lecture 21

The Digital Logic Level: Circuit Equivalence & Boolean Algebra

Dr. Barry O'Sullivan
`b.osullivan@cs.ucc.ie`

Course Homepage
`http://www.cs.ucc.ie/~osullb/cs1101`

Lecture Outline

- Circuit Equivalence
- Using only NAND and NOR Gates
- Laws of Boolean Algebra
- Identities of Boolean Algebra
- Consequences of DeMorgan's Law
- Using the Identities
- The EXCLUSIVE OR Gate
- Positive and Negative Logic
- **Reading**: Tanenbaum, Chapter 3, Section 1

Using only NAND and NOR Gates

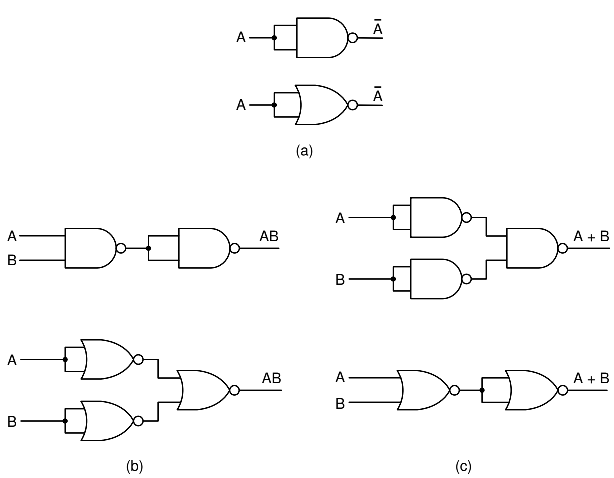


Figure 3-4: Construction of (a) NOT, (b) AND, and (c) OR gates using only NAND gates or only NOR gates.

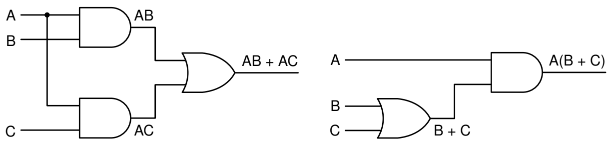
Circuit Equivalence

- Circuits with fewer and/or simpler gates (fewer inputs) are better.
- Boolean algebra can be a valuable tool for simplifying circuits.
- Example:

$$M = AB + AC$$

- Many of the rules of ordinary algebra also hold for Boolean algebra.
- In particular, $AB + AC$ can be factored into $A(B + C)$ using the distributive law.
- Two functions are equivalent if and only if they have the same output for all possible inputs
- Thus, $AB + AC$ is equivalent to $A(B + C)$.

Circuit Equivalence



A	B	C	AB	AC	AB + AC
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	0	0	0
1	0	1	0	1	1
1	1	0	1	0	1
1	1	1	1	1	1

(a)

A	B	C	A	B + C	A(B + C)
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	0	1	0
1	0	0	1	0	0
1	0	1	1	1	1
1	1	0	1	1	1
1	1	1	1	1	1

(b)

Figure 3-5. Two equivalent functions. (a) $AB + AC$ (b) $A(B + C)$.

Laws of Boolean Algebra

- In general, a circuit designer starts with a Boolean function and then apply the laws of Boolean algebra to it in an attempt to find a simpler but equivalent one.
- From the final function, a circuit can be constructed.
- To use this approach, we need some identities from Boolean algebra.

Identities of Boolean Algebra

Name	AND form	OR form
Identity law	1A = A	0 + A = A
Null law	0A = 0	1 + A = 1
Idempotent law	AA = A	A + A = A
Inverse law	$A\overline{A} = 0$	$A + \overline{A} = 1$
Commutative law	AB = BA	A + B = B + A
Associative law	(AB)C = A(BC)	(A + B) + C = A + (B + C)
Distributive law	A + BC = (A + B)(A + C)	A(B + C) = AB + AC
Absorption law	A(A + B) = A	A + AB = A
De Morgan's law	$\overline{AB} = \overline{A} + \overline{B}$	$\overline{A + B} = \overline{A}\,\overline{B}$

Figure 3-6. Some identities of Boolean algebra.

Comments on the Identities

- It is interesting to note that each law has two forms that are **duals** of each other.
- By interchanging AND and OR and also 0 and 1, either form can be produced from the other one.
- All the laws can be easily proven by constructing their truth tables.
- Except for DeMorgan's law, the absorption law, and the AND form of the distributive law, the results are reasonably intuitive.
- DeMorgan's law can be extended to more than two variables, for example, $\overline{ABC} = \overline{A} + \overline{B} + \overline{C}$.

Consequences of DeMorgan's Law

- DeMorgan's law suggests an alternative notation.
- An OR gate with inverted inputs is equivalent to a NAND gate.
- A NOR gate can be drawn as an AND gate with inverted inputs.
- Negating both forms of DeMorgan's law also has interesting consequences – leads to equivalent representations of the AND and OR gates.

Consequences of DeMorgan's Law

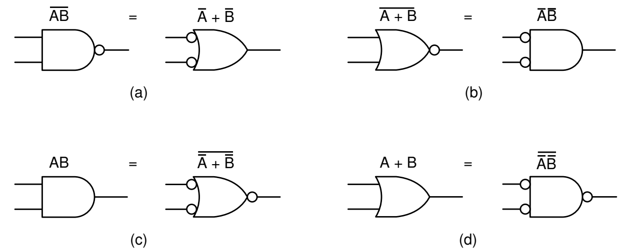


Figure 3-7. Alternative symbols for some gates: (a) NAND. (b) NOR. (C) AND. (d) OR.

Using the Identities

- Using the identities it is easy to convert the sum-of-products representation of a truth table to pure NAND or pure NOR form.
- Example: consider the EXCLUSIVE OR function:

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

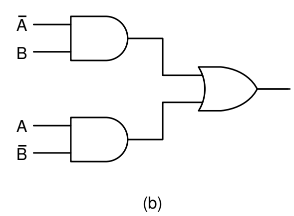
- How do we get convert this to a completely NAND form?
- The standard sum-of-products circuit is shown in Fig. 3-8(b). To
- Note that inversion bubbles can be moved along a line at will

CS1101: Systems Organisation *The Digital Logic Level*

The EXCLUSIVE OR Gate

A	B	XOR
0	0	0
0	1	1
1	0	1
1	1	0

(a)



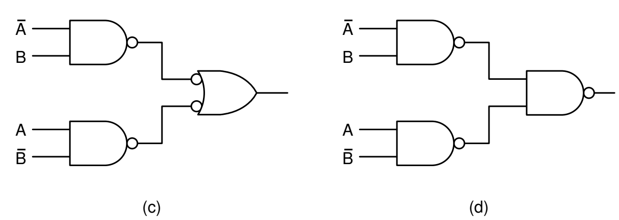


Figure 3-8. (a) The truth table for the XOR function. (b)-(d) Three circuits for computing it.

Positive and Negative Logic

- As a final note on circuit equivalence, we will now demonstrate the surprising result that the same physical gate can compute different functions, depending on the conventions used.
- If we adopt the convention that 0 volts is logical 0 and 5 volts is logical 1, this is called **positive logic**.
- If, however, in **negative logic**, 0 volts denotes a logical 1 and 5 volts a logical 0.
- What is the significance?

Positive and Negative Logic

A	**B**	**F**
0^V	0^V	0^V
0^V	5^V	0^V
5^V	0^V	0^V
5^V	5^V	5^V

(a)

A	**B**	**F**
0	0	0
0	1	0
1	0	0
1	1	1

(b)

A	**B**	**F**
1	1	1
1	0	1
0	1	1
0	0	0

(c)

Figure 3-9. (a) Electrical characteristics of a device. (b) Positive logic. (c) Negative logic.

- Thus, the convention chosen to map voltages onto logical values is critical.
- Except where otherwise specified, we will henceforth use positive logic, so the terms logical 1, true, and high are synonyms, as are logical 0, false, and low.