Logic Gates

Objectives

- Identify the basic gates and describe the behavior of each
- Combine basic gates into circuits
- Describe the behavior of a gate using Boolean expressions, truth tables, and logic diagrams

Definition

- A **gate** is a device that performs a basic operation on electrical signals
- Gates are combined into **circuits** to perform more complicated tasks
- describing the behavior of gates
  and circuits by
  - Boolean expressions
  - logic diagrams
  - truth tables

Gates

six types of gates

- NOT
- AND
- OR
- XOR
- NAND
- NOR

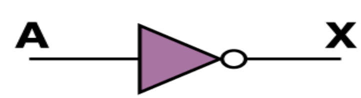
NOT Gate

- A NOT gate accepts one input value and produces one output value
- By definition, if the input value for a NOT gate is 0, the output value is 1, and if the input value is 1, the output is 0
- A NOT gate is sometimes referred to as an *inverter* because it inverts the input value

Boolean Expression

$$X = A'$$

Logic Diagram Symbol



Truth Table

A	X
0	1
1	0

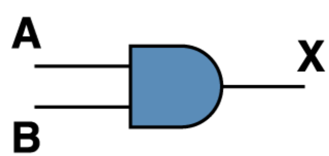
AND Gate

- An AND gate accepts two input signals
- If the two input values for an AND gate are both 1, the output is 1; otherwise, the output is 0

Boolean Expression

$$X = A \cdot B$$

Logic Diagram Symbol



Truth Table

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

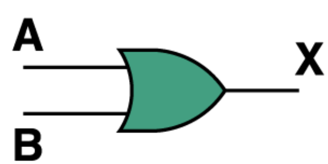
OR Gate

- If the two input values are both 0, the output value is 0; otherwise, the output is 1

Boolean Expression

$$X = A + B$$

Logic Diagram Symbol



Truth Table

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

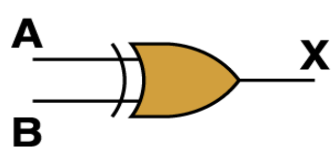
XOR Gate

- XOR, or *exclusive* OR, gate
  - An XOR gate produces 0 if its two inputs are the same, and a 1 otherwise
  - Note the difference between the XOR gate and the OR gate; they differ only in one input situation
  - When both input signals are 1, the OR gate produces a 1 and the XOR produces a 0

Boolean Expression

$$X = A \oplus B$$

Logic Diagram Symbol



Truth Table

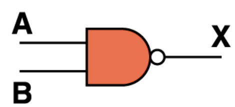
A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

NAND and NOR Gates

- The NAND and NOR gates are essentially the opposite of the AND and OR gates, respectively

Boolean Expression | **Logic Diagram Symbol** | **Truth Table**

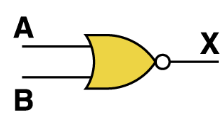
X = (A · B)'



A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

Boolean Expression | **Logic Diagram Symbol** | **Truth Table**

X = (A + B)'



A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

Review of Gate Processing

- A NOT gate inverts its single input value
- An AND gate produces 1 if both input values are 1
- An OR gate produces 1 if one or the other or both input values are 1
- An XOR gate produces 1 if one or the other (but not both) input values are 1
- A NAND gate produces the opposite results of an AND gate
- A NOR gate produces the opposite results of an OR gate

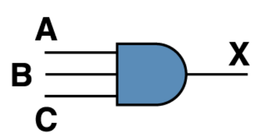
Gates with More Inputs

- Gates can be designed to accept three or more input values
- A three-input AND gate, for example, produces an output of 1 only if all input values are 1

Boolean Expression

$X = A \cdot B \cdot C$

Logic Diagram Symbol



Truth Table

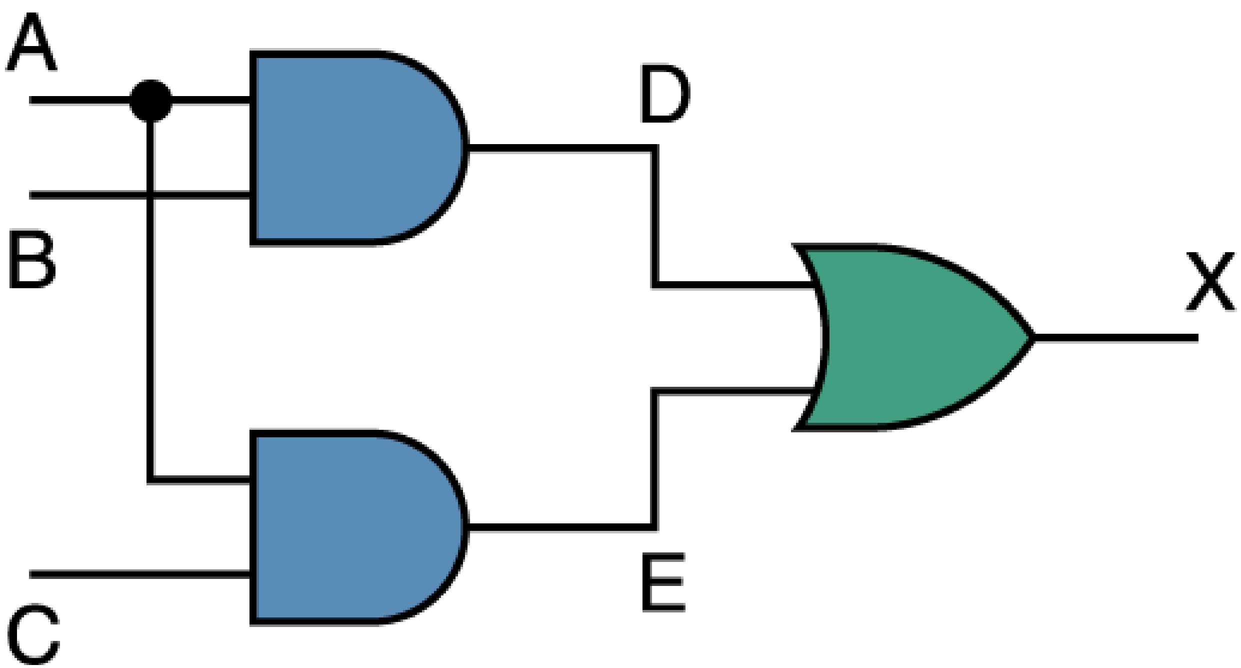
A	B	C	X
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Circuits

- Two general categories
  - In a **combinational circuit**, the input values explicitly determine the output
  - In a **sequential circuit**, the output is a function of the input values as well as the existing state of the circuit
- As with gates, we can describe the operations of entire circuits using three notations
  - Boolean expressions
  - logic diagrams
  - truth tables

Combinational Circuits

- Gates are combined into circuits by using the output of one gate as the input for another



Combinational Circuits

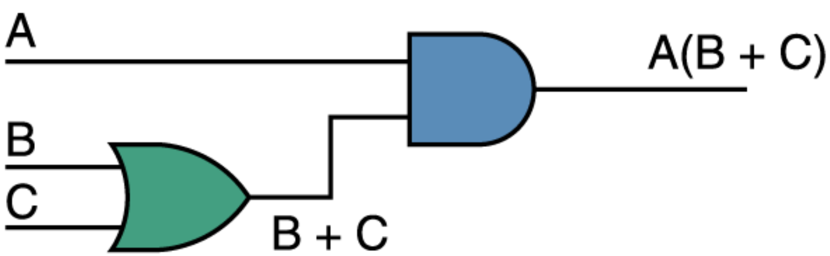
A	B	C	D	E	X
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	0	0	0
1	0	1	0	1	1
1	1	0	1	0	1
1	1	1	1	1	1

- Because there are three inputs to this circuit, eight rows are required to describe all possible input combinations
- This same circuit using Boolean algebra:

  (AB + AC)

Now let's go the other way; let's take a Boolean expression and draw

- Consider the following Boolean expression: $A(B + C)$



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

- Now compare the final result column in this truth table to the truth table for the previous example
  - They are identical

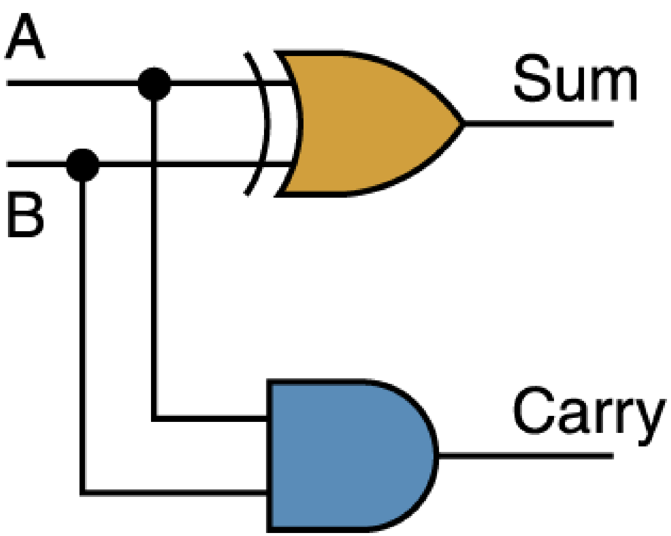
Properties of Boolean Algebra

Property	AND	OR
Commutative	AB = BA	A + B = B + A
Associative	(AB)C = A(BC)	(A + B) + C = A + (B + C)
Distributive	A(B + C) = (AB) + (AC)	A + (BC) = (A + B)(A + C)
Identity	A1 = A	A + 0 = A
Complement	A(A') = 0	A + (A') = 1
DeMorgan's law	(AB)' = A' OR B'	(A + B)' = A'B'

Adders

- At the digital logic level, addition is performed in binary
- Addition operations are carried out
  by special circuits called, appropriately, **adders**
- The result of adding two binary digits could produce
  a *carry value*
- Recall that 1 + 1 = 10
  in base two
- A circuit that computes the sum of two bits
  and produces the correct carry bit is called a **half adder**

Adders



- Circuit diagram representing a half adder
- Two Boolean expressions:

$$\text{sum} = A \oplus B$$
$$\text{carry} = AB$$

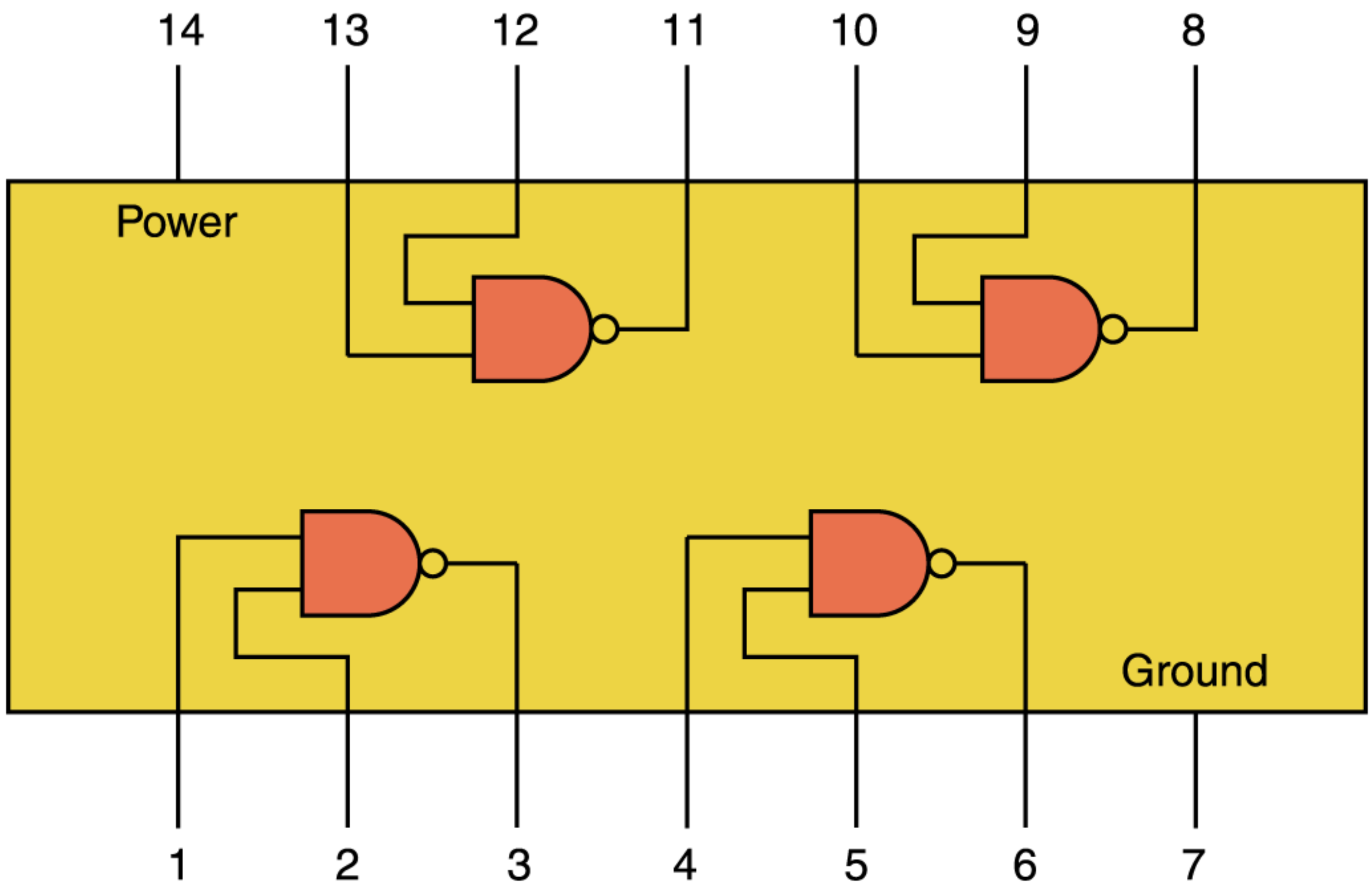
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Integrated Circuits

- Integrated circuits (IC) are classified by the number of gates contained in them

Abbreviation	Name	Number of Gates
SSI	Small-Scale Integration	1 to 10
MSI	Medium-Scale Integration	10 to 100
LSI	Large-Scale Integration	100 to 100,000
VLSI	Very-Large-Scale Integration	more than 100,000

Integrated Circuits



CPU Chips

- The most important integrated circuit
  in any computer is the Central Processing Unit, or CPU
- Each CPU chip has a large number
  of pins through which essentially all communication in a computer system occurs