computer science illuminated

Gates and Circuits

Nell Dale & John Lewis
(adaptation by Erin Chambers and Michael Goldwasser)

Computers and Electricity

- A **gate** is a device that performs a basic operation on electrical signals
- Gates are combined into **circuits** to perform more complicated tasks

Gates

- Let's examine the processing of the following six types of gates
  - NOT
  - AND
  - OR
  - XOR
  - NAND
  - NOR

Describing Gates and Circuits

- There are three different, but equally powerful, notational methods
  for describing the behavior of gates and circuits
  - Boolean expressions
  - logic diagrams
  - truth tables

Describing Gates and Circuits

- **Boolean algebra**: expressions in this algebraic notation are an elegant and powerful way to demonstrate the activity of electrical circuits
  - Basic propositional statements are unambiguously either True or False
  - Operations such as AND or NOT are then performed on these values
  - A gate is simply a mechanical way to perform such a boolean operation

Describing Gates and Circuits

- **Logic diagram:** a graphical representation of a circuit
  - Each type of gate is represented by a specific graphical symbol
- **Truth table:** defines the function of a gate by listing all possible input combinations that the gate could encounter, and the corresponding output

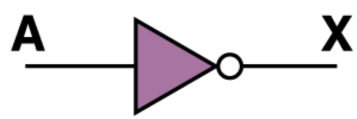
NOT Gate

- a NOT gate accepts one input value and produces one output value

Boolean Expression

$$X = A'$$

Logic Diagram Symbol



Truth Table

A	X
0	1
1	0

- a NOT gate is sometimes referred to as an *inverter* because it inverts the input value

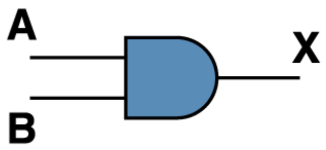
AND Gate

- 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

Figure 4.2 Various representations of an AND gate

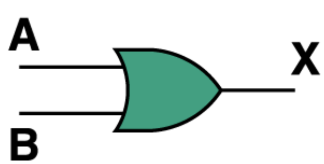
OR Gate

- 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

Figure 4.3 Various representations of a OR gate

XOR Gate

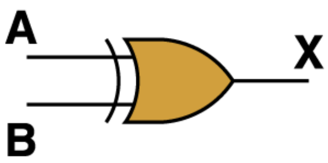
- XOR, or *exclusive* OR, gate
  - An XOR gate produces 0 if its two inputs are the same, and a 1 otherwise
  - Note the difference between the XOR gate and the OR gate; they differ only in one input situation
  - When both input signals are 1, the OR gate produces a 1 and the XOR produces a 0

XOR Gate

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

Figure 4.4 Various representations of an XOR gate

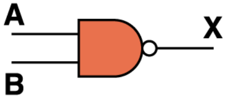
NAND and NOR Gates

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

Boolean Expression

$$X = (A \cdot B)'$$

Logic Diagram Symbol



Truth Table

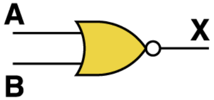
A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

Figure 4.5 **Various representations of a NAND gate**

Boolean Expression

$$X = (A + B)'$$

Logic Diagram Symbol



Truth Table

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

Figure 4.6 **Various representations of a NOR gate**

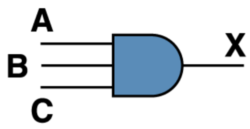
Gates with More Inputs

- 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

Figure 4.7 Various representations of a three-input AND gate

Constructing Gates

- A **transistor** is a device that acts, depending on the voltage level of an input signal, either as a wire that conducts electricity or as a resistor that blocks the flow of electricity
  - A transistor has no moving parts, yet acts like a switch
  - It is made of a **semiconductor** material, which is neither a particularly good conductor of electricity, such as copper, nor a particularly good insulator, such as rubber

jasonm:

Redo 4.8 (crop)

Constructing Gates

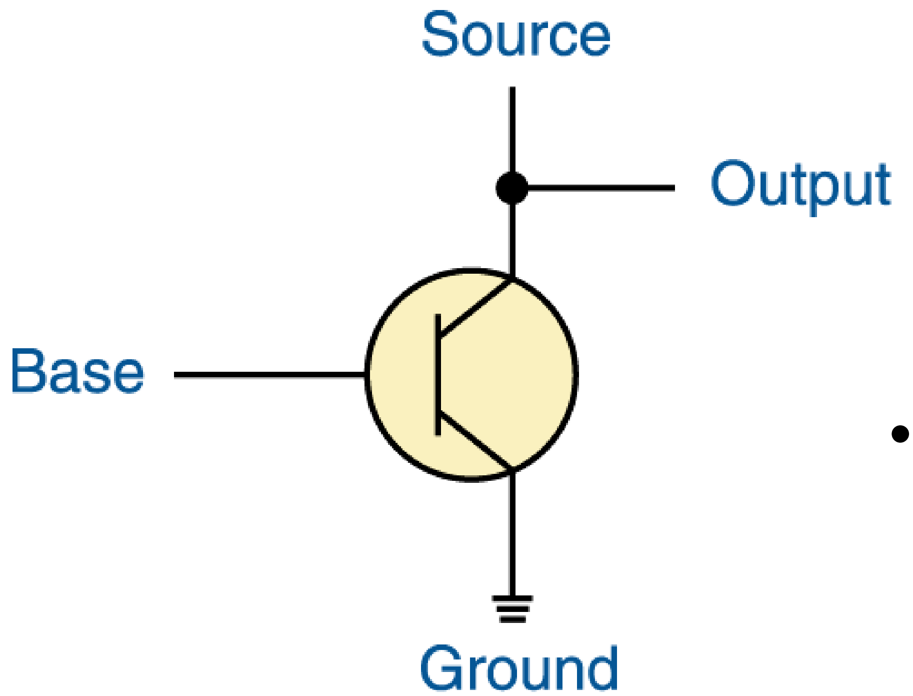


Figure 4.8 **The connections of a transistor**

- A transistor has three terminals
  - A source
  - A base
  - An emitter, typically connected to a ground wire
- If the electrical signal is grounded, it is allowed to flow through an alternative route to the ground (literally) where it can do no harm

Constructing Gates

- It turns out that, because the way a transistor works, the easiest gates to create are the NOT, NAND, and NOR gates

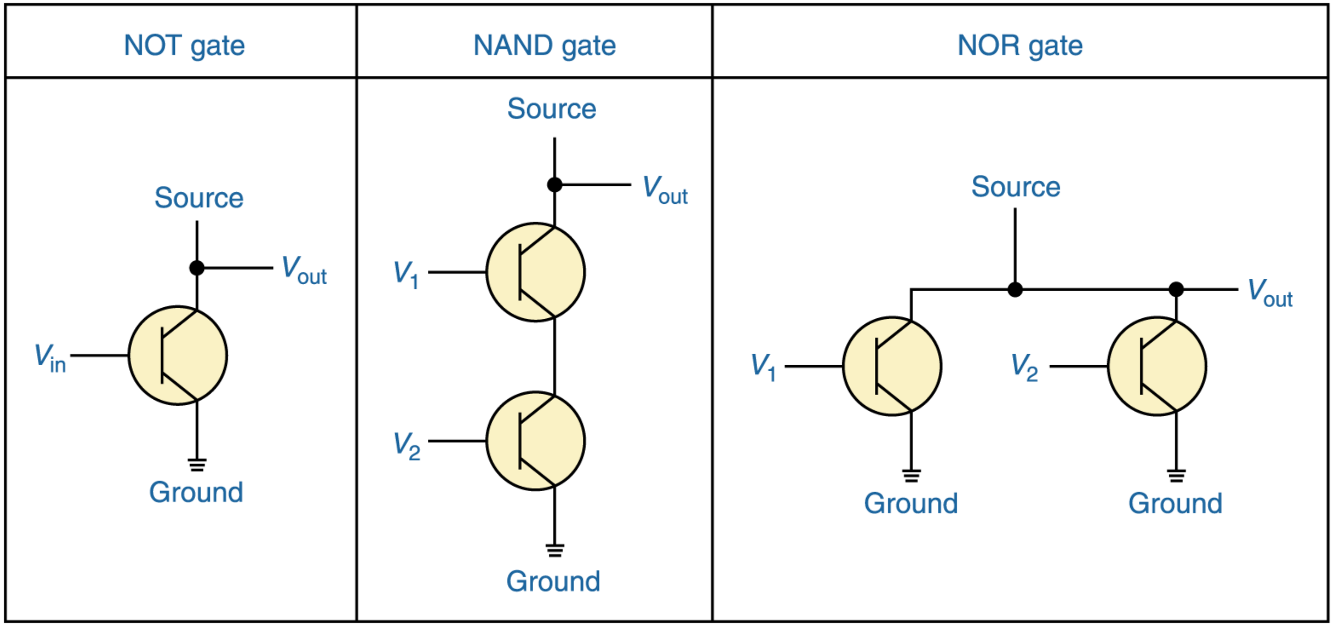


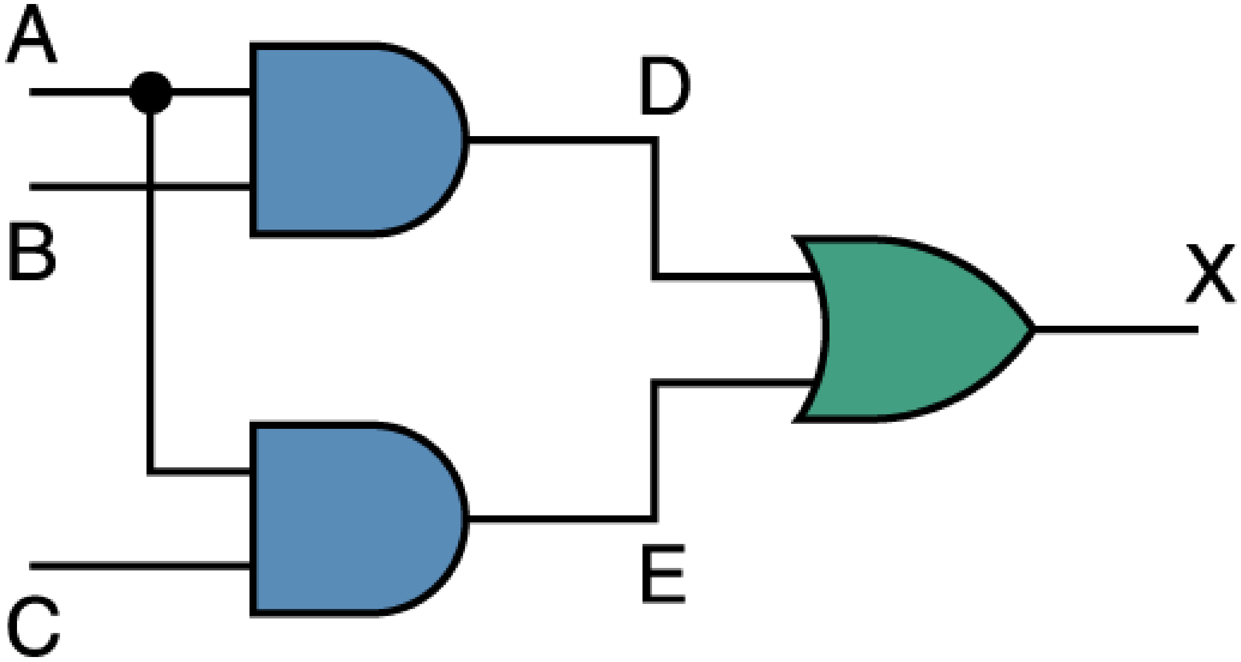
Figure 4.9 **Constructing gates using transistors**

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



jasonm:

Redo to get white space around table (p100)

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

Page 100

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

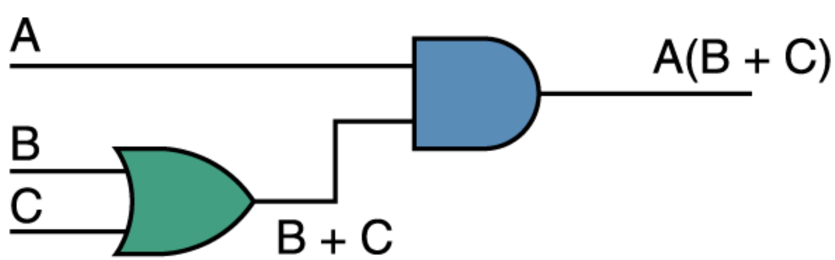
$$(AB + AC)$$

jasonm:

Redo table to get white space (p101)

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

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



Page 100

Page 101

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

jasonm:

Redo table (p101)

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'

Page 101

Adders

- At the digital logic level, addition is performed in binary
- Addition operations are carried out by special circuits called, appropriately, **adders**

jasonm:

Redo table (p103)

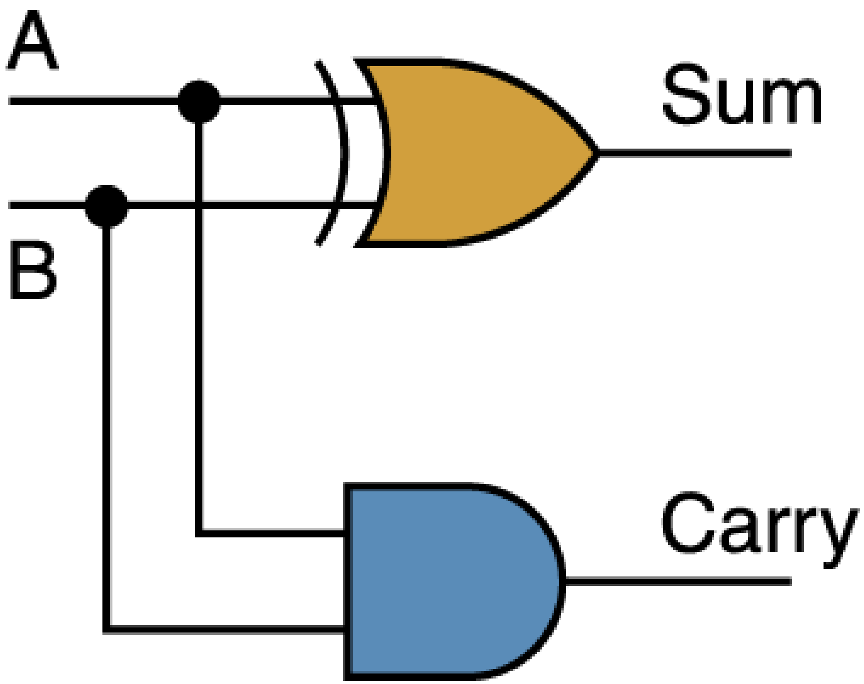
Adders

- 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**

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

Page 103

Adders



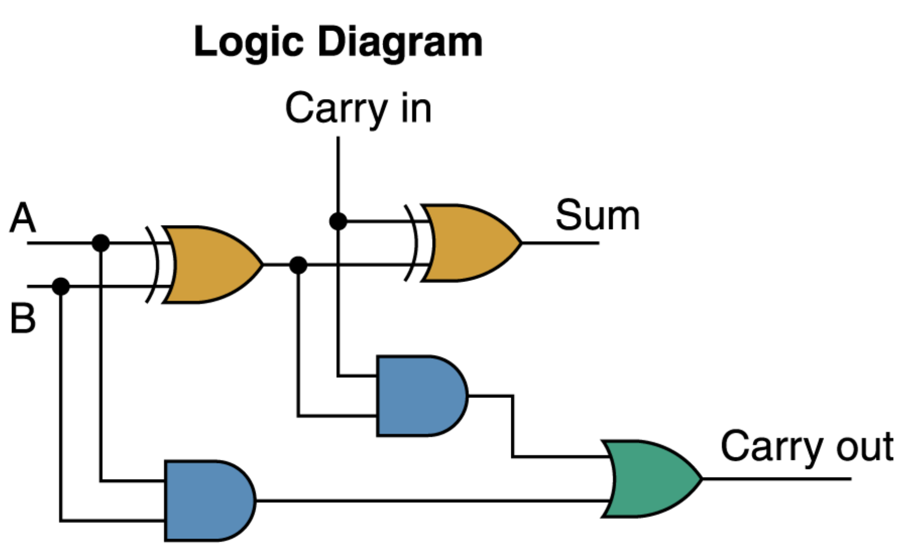
- Circuit diagram representing a half adder
- Two Boolean expressions:

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

Page 103

Adders

- A circuit called a **full adder** takes the carry-in value into account



Truth Table

A	B	Carry-in	Sum	Carry-out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Figure 4.10 **A full adder**

Multiplexers

- **Multiplexer** is a general circuit that produces a single output signal
  - The output is equal to one of several input signals to the circuit
  - The multiplexer selects which input signal is used as an output signal based on the value represented by a few more input signals, called *select signals* or *select control lines*

Multiplexers

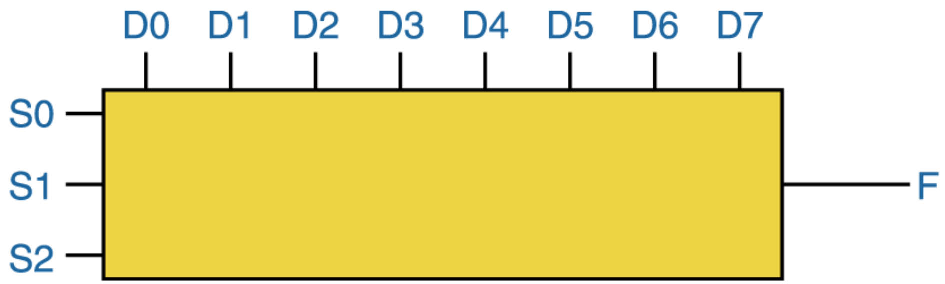


Figure 4.11 A block diagram of a multiplexer with three select control lines

S0	S1	S2	F
0	0	0	D0
0	0	1	D1
0	1	0	D2
0	1	1	D3
1	0	0	D4
1	0	1	D5
1	1	0	D6
1	1	1	D7

- The control lines S0, S1, and S2 determine which of eight other input lines (D0 through D7) are routed to the output (F)

Circuits as Memory

- Digital circuits can be used to store information
- These circuits form a sequential circuit, because the output of the circuit is also used as input to the circuit

Circuits as Memory

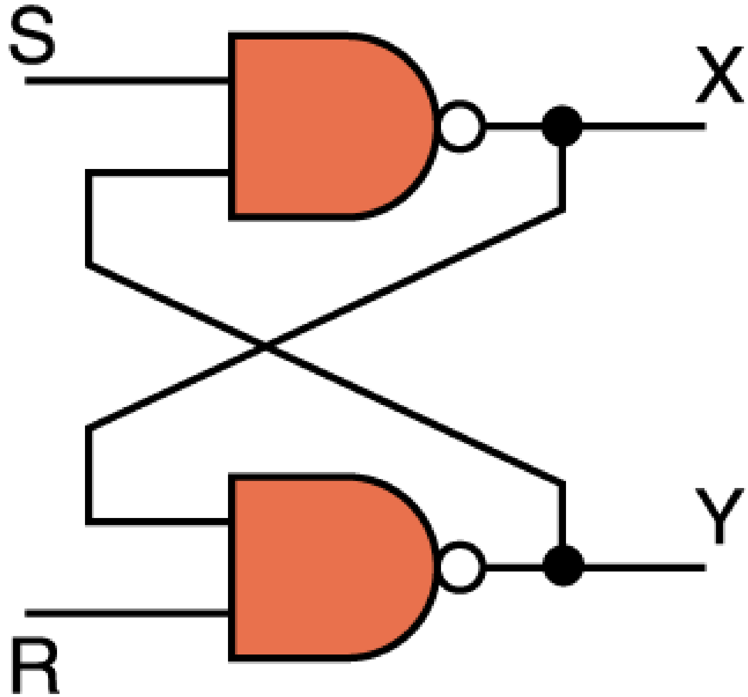


Figure 4.12 **An S-R latch**

- An S-R latch stores a single binary digit (1 or 0)
- There are several ways an S-R latch circuit could be designed using various kinds of gates

Circuits as Memory

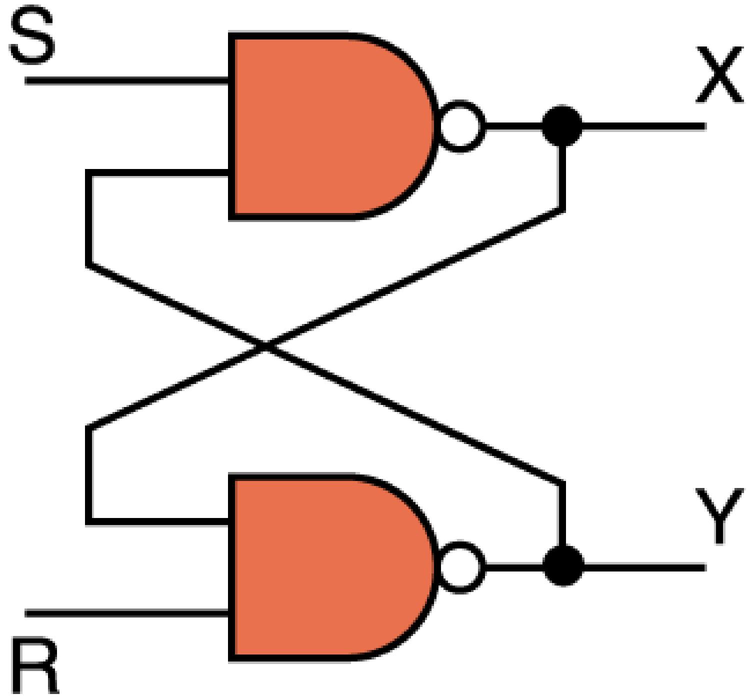


Figure 4.12 **An S-R latch**

- The design of this circuit guarantees that the two outputs X and Y are always complements of each other
- The value of X at any point in time is considered to be the current state of the circuit
- Therefore, if X is 1, the circuit is storing a 1; if X is 0, the circuit is storing a 0

Integrated Circuits

- An **integrated circuit** (also called a *chip*) is a piece of silicon on which multiple gates have been embedded
- These silicon pieces are mounted on a plastic or ceramic package with pins along the edges that can be soldered onto circuit boards or inserted into appropriate sockets

jasonm:

Redo table (p107)

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

Page 107

Integrated Circuits

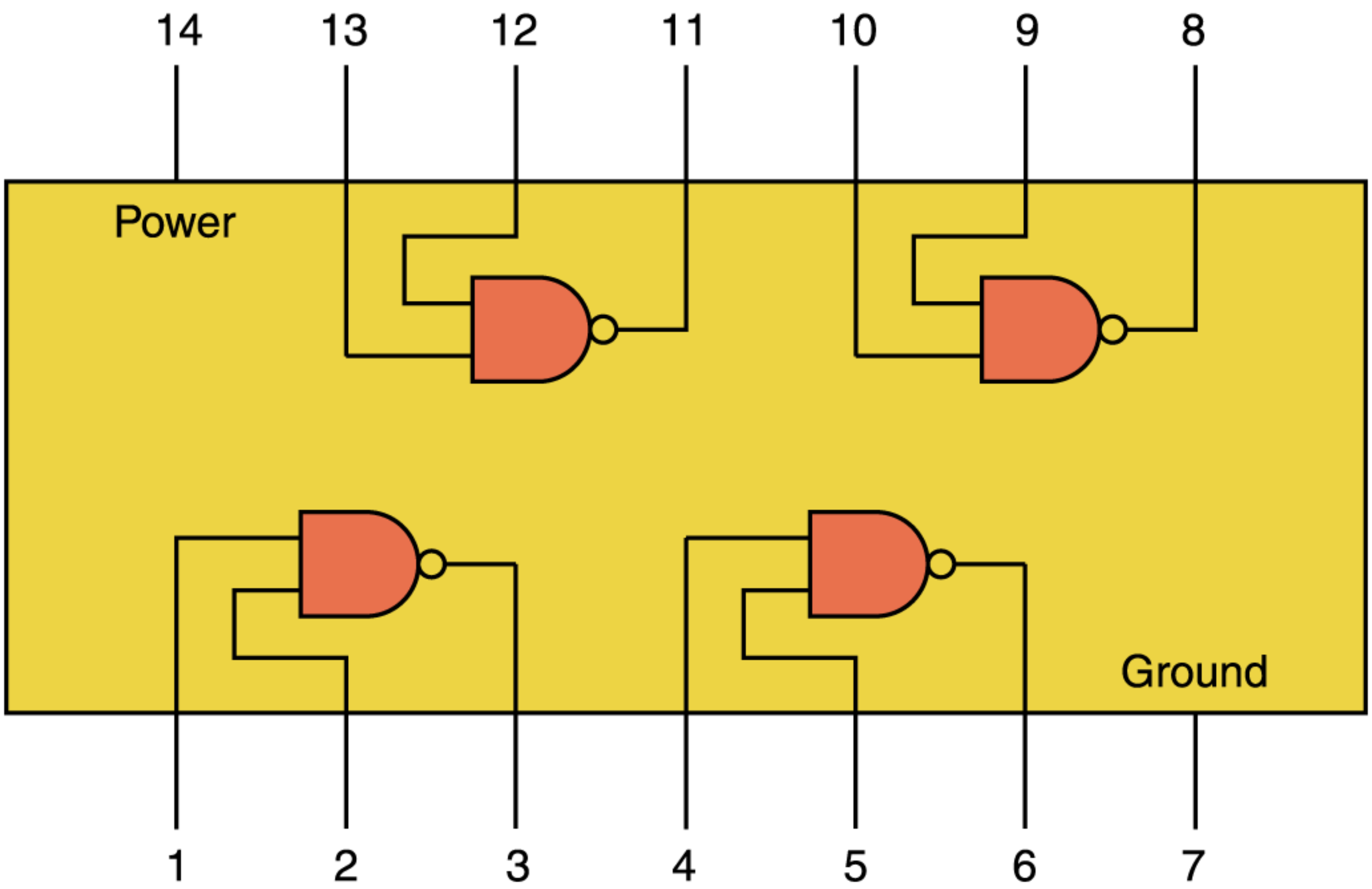


Figure 4.13 An SSI chip contains independent NAND gates

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