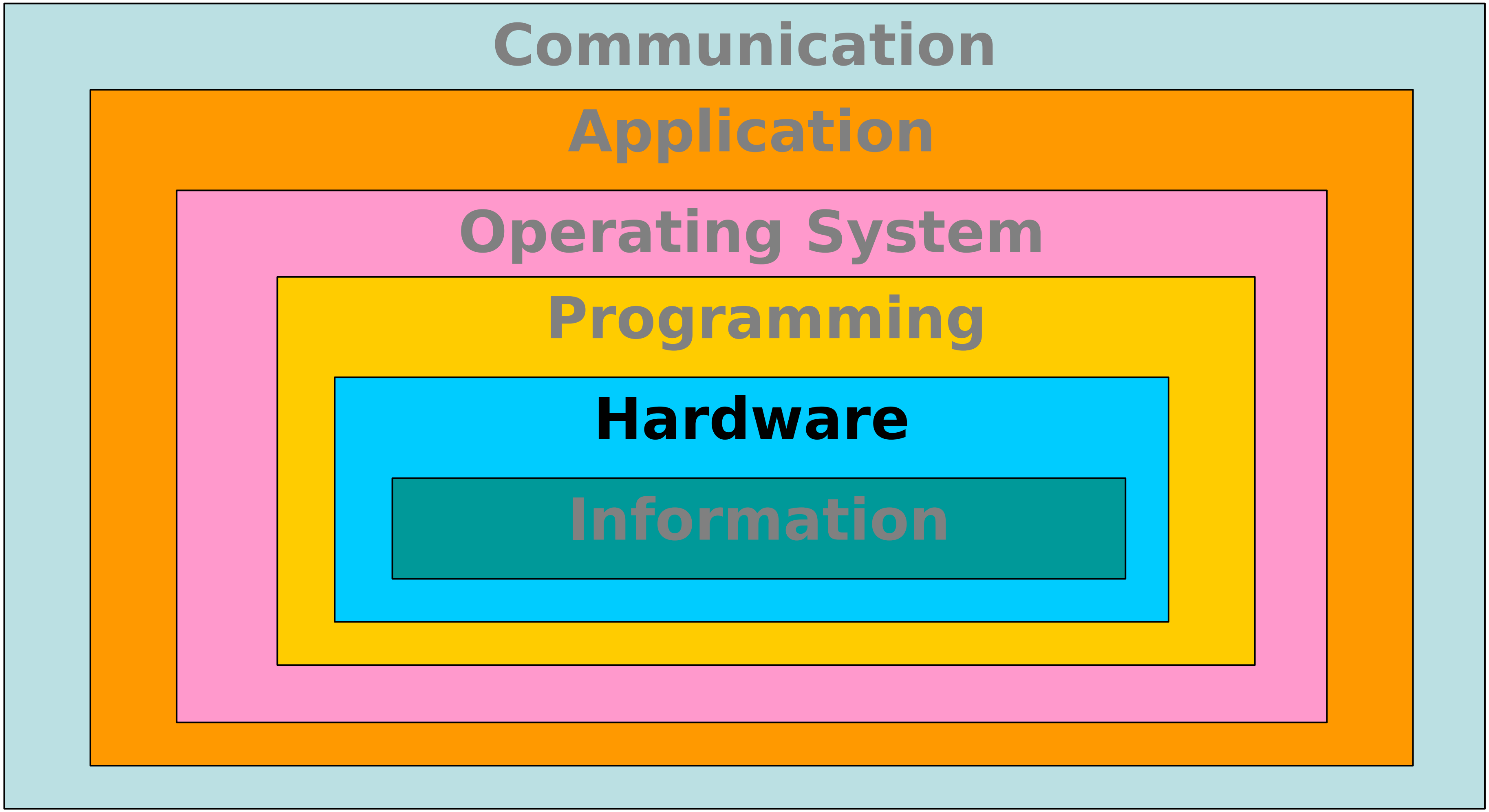
Chapter 4

Gates and Circuits

Layers of a Computing System



Chapter Goals

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

Computers and Electricity

- **Gate** - A device that performs a basic operation on electrical signals
- **Circuits** - Gates combined to perform more complicated tasks

Computers and Electricity

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

Computers and Electricity

- **Boolean expressions** Expressions in Boolean algebra, a mathematical notation for expressing two-valued logic

  This algebraic notation is an elegant and powerful way to demonstrate the activity of electrical circuits

Computers and Electricity

- **Logic diagram** A graphical representation of a circuit
  - Each type of gate is represented by a specific graphical symbol
- **Truth table** A table showing all possible input value and the associated output values

Gates

- Let's examine the processing of the following six types of gates
  - **NOT**
  - **AND**
  - **OR**
  - **XOR**
  - **NAND**
  - **NOR**
- Typically, logic diagrams are black and white, and the gates are distinguished only by their shape

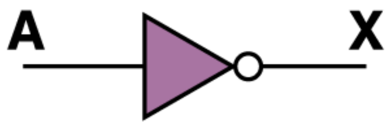
NOT Gate

- 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

Figure 4.1 Various representations of a NOT gate

NOT Gate

- 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

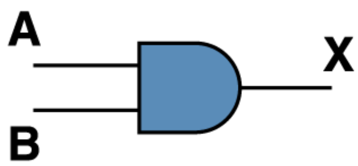
AND Gate

- 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**

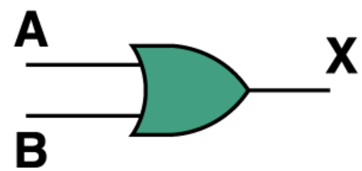
OR Gate

- 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

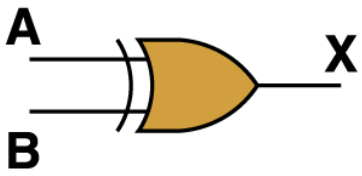
- **XOR**, or *exclusive* OR, gate
  - An XOR gate produces 0 if its two inputs are the same, and a 1 otherwise
  - Note the difference between the XOR gate and the OR gate; they differ only in one input situation
  - When both input signals are 1, the OR gate produces a 1 and the XOR produces a 0

XOR Gate

Boolean Expression

$$X = A \oplus B$$

Logic Diagram Symbol



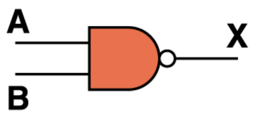
Truth Table

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

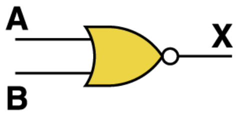
- The **NAND** and **NOR** gates are essentially the opposite of the AND and OR gates, respectively

Boolean Expression	Logic Diagram Symbol
X = (A · B)'	

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	Logic Diagram Symbol
X = (A + B)'	

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**

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

Review of Gate Processing

- 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

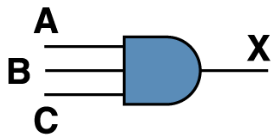
Gates with More Inputs

- 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

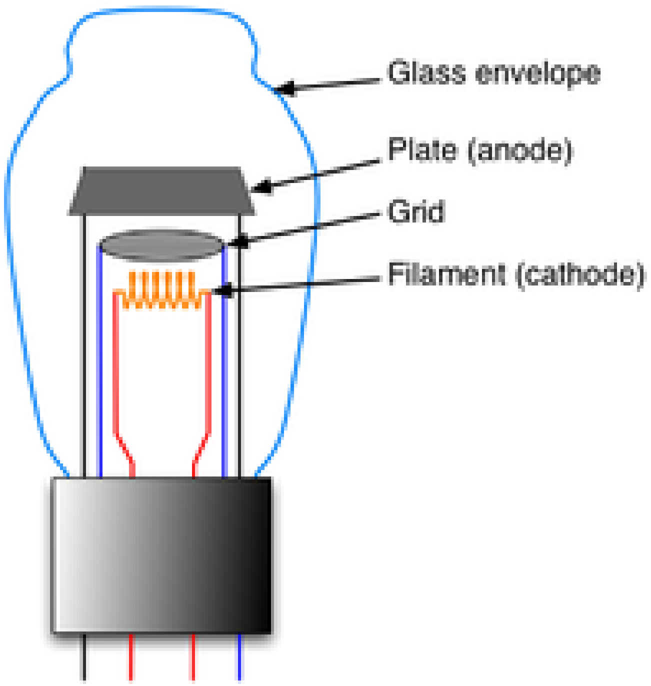
A	B	C	X
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

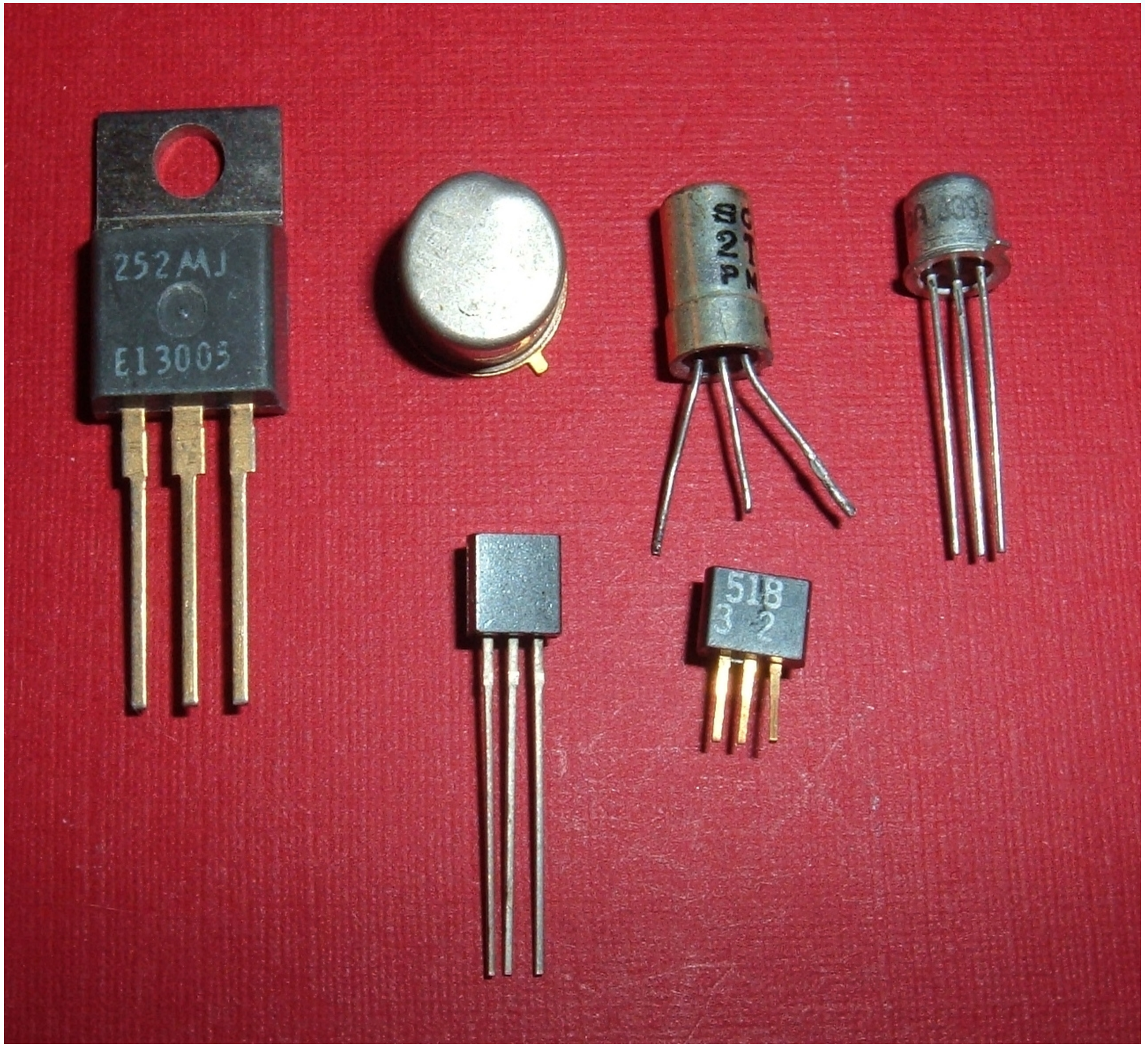
Figure 4.7 Various representations of a three-input AND gate

Constructing Gates

- **Transistor** 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

Tube & Transistors





Constructing Gates

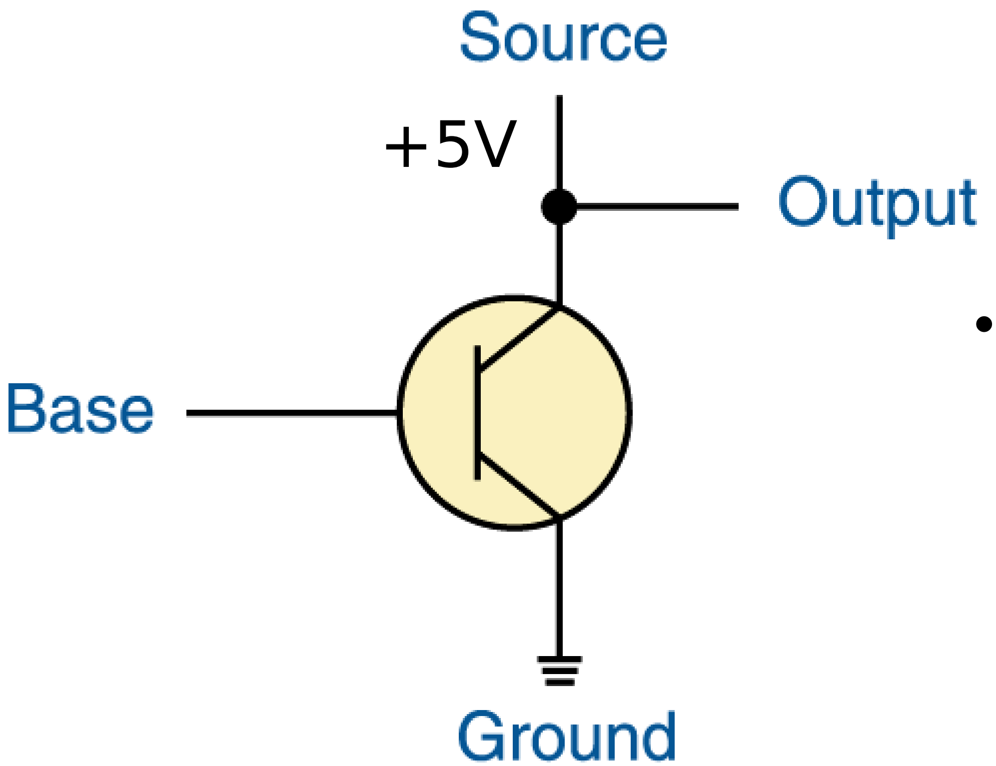


Figure 4.8 **The connections of a transistor**

- A transistor has three terminals
  - A **source**
  - A **base**
  - An **emitter**, typically connected to a ground wire
- If the electrical signal is grounded (base is high), it is allowed to flow through an alternative route to the ground (literally) where it can do no harm (source is low), otherwise source is high (+5V)

Constructing Gates

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

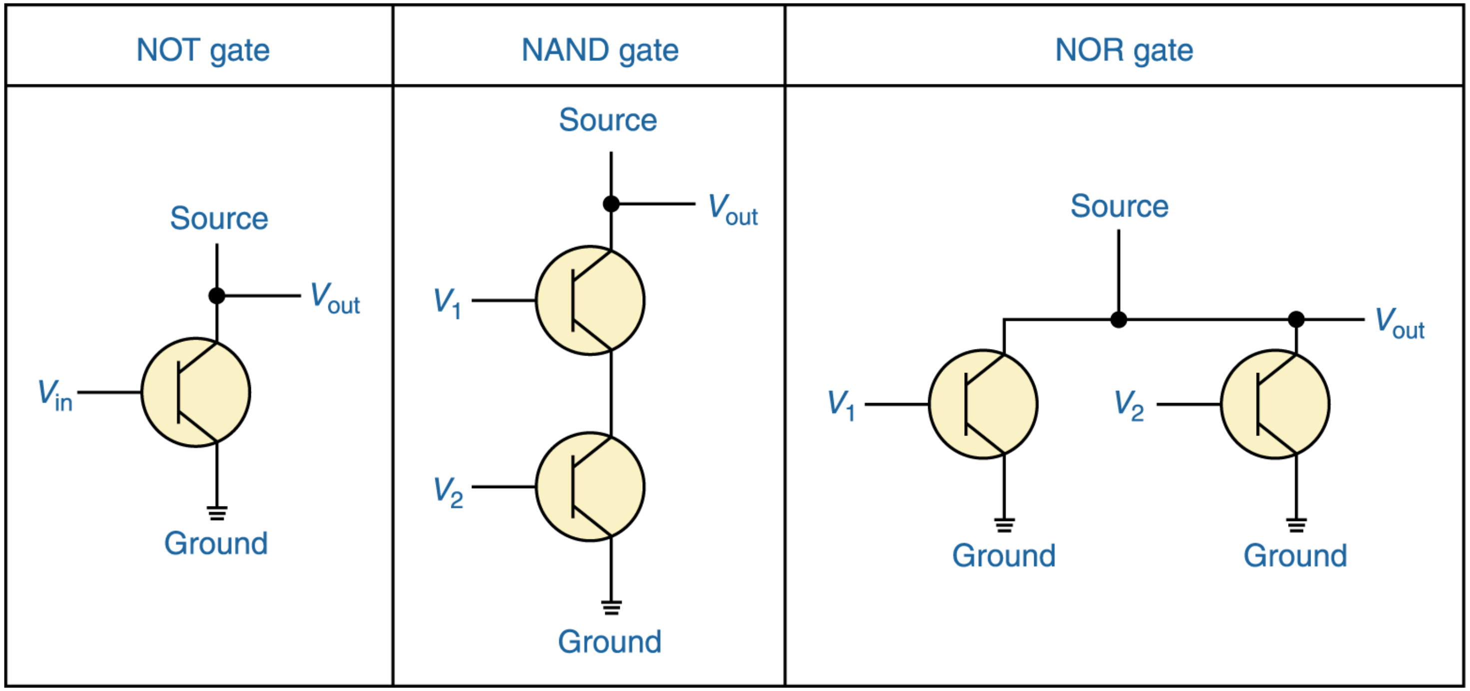
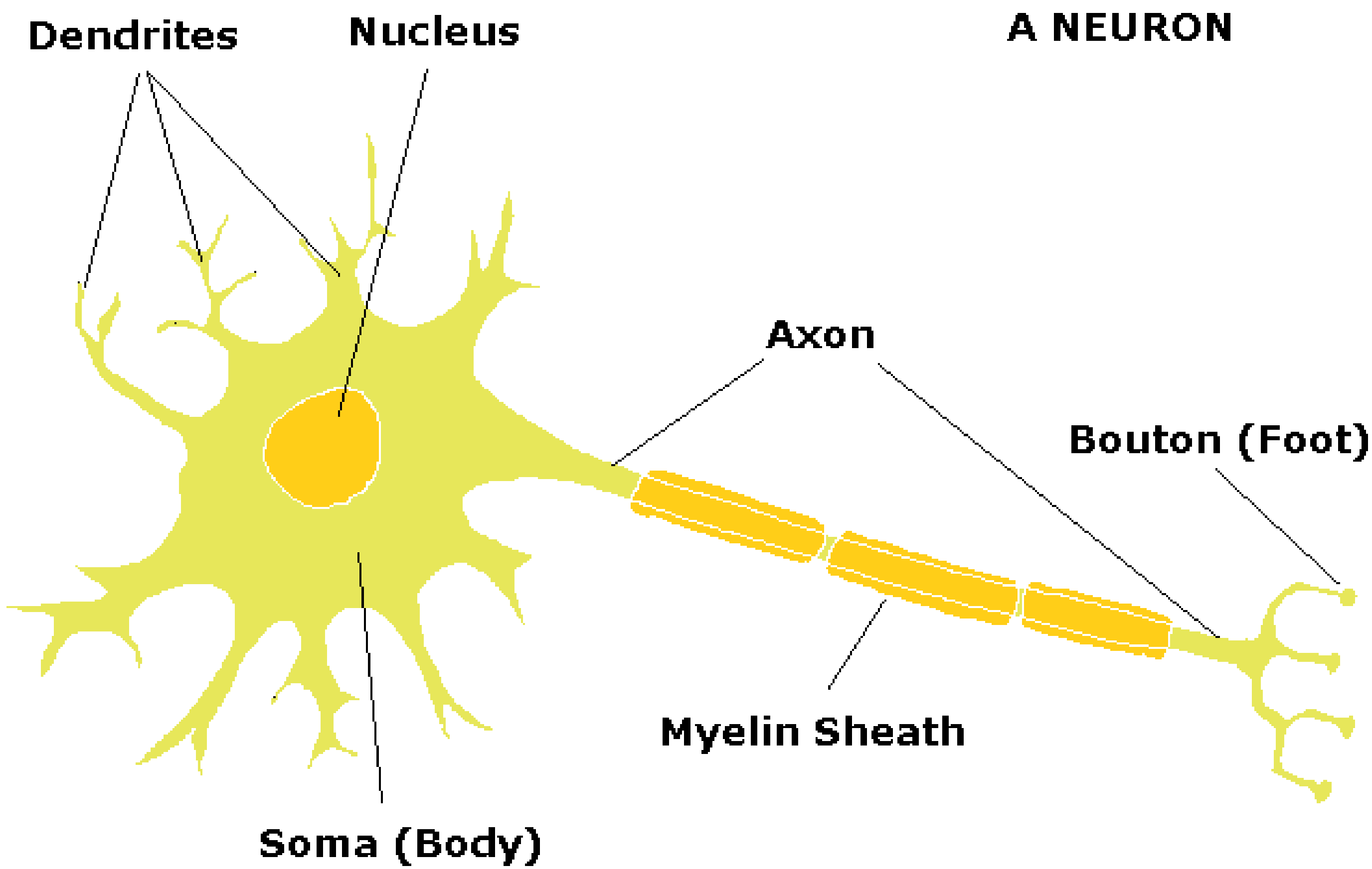


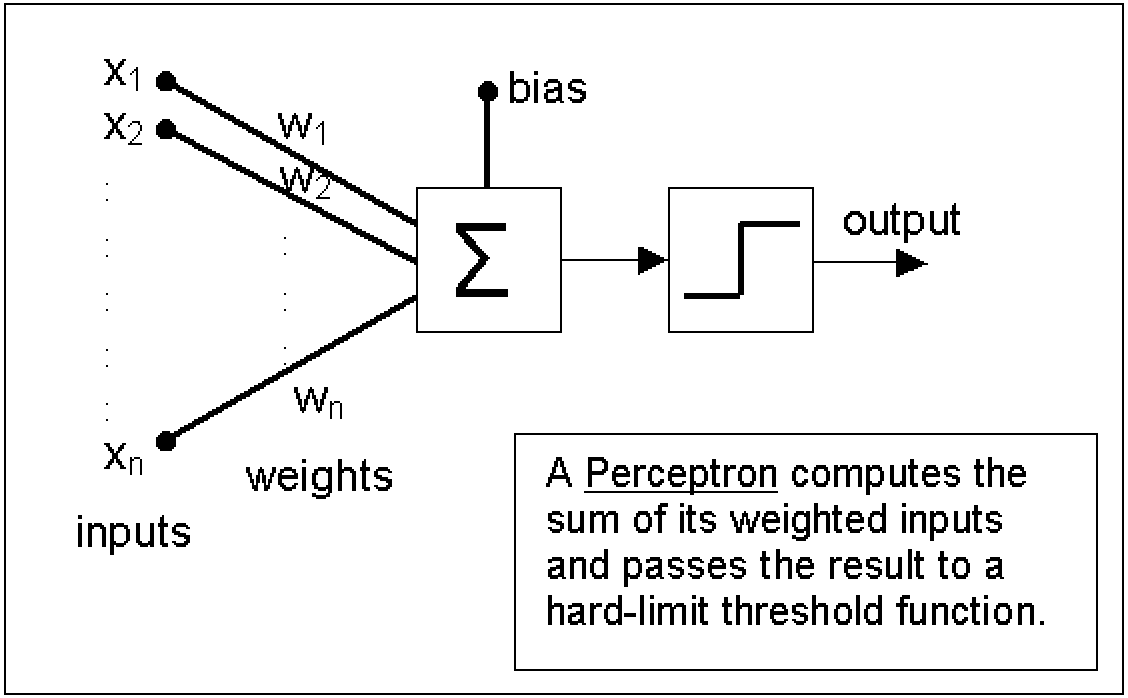
Figure 4.9 **Constructing gates using transistors**

Getting Personal



- McCulloch & Pitts – 1943, Artificial Neurons

Perceptron



- Rosenblatt, 1962 – Perceptron Learning paper generates great interest
- Minsky and Papert, 1969 – Computational Geometry proof starts a 20 year hiatus in Artificial Intelligence

"my brain hurts!"

Homework

- **Read Chapter Four, Sections 4.1 – 4.3**
- **Exercise: P117, 18-29**

Assignment One

- **Due Today, Wednesday**
  - No lateness

Next Class

- **No** Class Monday
- Next Class is Wednesday, 10/4
- Have a Nice Weekend...