The Building Blocks: Binary Numbers, Boolean Logic, and Gates

Chapter 4

- ➢Representing Information
- ➢The Binary Numbering System
- ➢Boolean Logic and Gates
- ➢Building Computer Circuits
- ➢Control Circuits

Purpose of Chapter

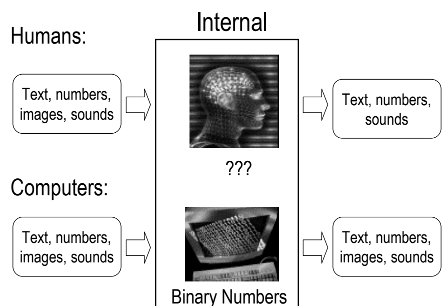
- Learn how computers represent and store information.
- Learn why computers represent information that way.
- Learn what the basic building devices in a computer are, and how those devices are used to store information.
- Learn how to build more complex devices using the basic devices.

External Representation of Information

- When we communicate with each other, we need to represent the information in an understandable notation, e.g.
  - We use digits to represent numbers.
  - We use letters to represent text.
- Same applies when we communicate with a computer:
  - We enter text and numbers on the keyboard,
  - The computers displays text, images, and numbers on the screen.
- We refer to this as an <u>external</u> representation.
  - But how do humans/computers store the information "internally"?

Internal Representation of Information

- Humans:
- Computers:



What information do we need to represent?

- Numbers
  - Integers (234, 456)
  - Positive/negative value (-100, -23)
  - Floating point numbers (12.345, 3.14159)
- Text
  - Characters (letters, digits, symbols)
- Other
  - Graphics, Sound, Video, …

Numbering Systems

- We use the decimal numbering system
  - 10 digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
  - For example: 12
- Why use 10 digits (symbols)?
  - Roman: I (=1) V (=5) X (=10) L (=50), C(=100)
  - XII = 12, Pentium III
- What if we only had one symbol?
  - IIIII IIIII II = 12
- What system do computers use?

The Binary Numbering System

- All computers use the binary numbering system
  - Only two digits: 0, 1
  - For example: 10, 10001, 10110
- Similar to decimal, except uses a different base
  - Binary (base-2): 0, 1
  - Decimal (base-10): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
  - Octal (base-8): 0, 1, 2, 3, 4, 5, 6, 7
  - Hexadecimal (base-16):
    - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F (A=10, ..., F=15)
- What do we mean by a base?

CMPUT101 Introduction to Computing (c) Yngvi Bjornsson 7

Decimal vs. Binary Numbers

- What does the decimal value 163 stand for?

	10^2	10^1	10^0
163	1	6	3
	1x100 +	6x10 +	3x1

base

- What does the binary value 101 stand for?

	2^2	2^1	2^0
101	1	0	1
	1x4 +	0x2 +	1x1

base

CMPUT101 Introduction to Computing (c) Yngvi Bjornsson 8

Binary-to-Decimal Conversion Table

Decimal	Binary	Decimal	Binary	Decimal	Binary	Decimal	Binary
0	0	8	1000	16	10000	24	11000
1	1	9	1001	17	10001	25	11001
2	10	10	1010	18	10010	26	11010
3	11	11	1011	19	10011	27	11011
4	100	12	1100	20	10100	28	11100
5	101	13	1101	21	10101	29	11101
6	110	14	1110	22	10110	30	11110
7	111	15	1111	23	10111	31	11111

CMPUT101 Introduction to Computing (c) Yngvi Bjornsson 9

Converting from Binary to Decimal

- What is the decimal value of the binary value 101 ?

	2^2	2^1	2^0	
101	1	0	1	
	1x4 +	0x2 +	1x1	
	4 +	0 +	1	= 5

- What is the decimal value of the binary value 1110 ?

	2^3	2^2	2^1	2^0	
1110	1	1	1	0	
	1x8 +	1x4 +	1x2 +	0x1	
	8 +	4 +	2 +	0	= 14

CMPUT101 Introduction to Computing (c) Yngvi Bjornsson 10

Bits

- The two binary digits 0 and 1 are frequently referred to as bits.
- How many bits does a computer use to store an integer?
  - Intel Pentium PC = 32 bits
  - Alpha = 64 bits
- What if we try to compute a larger integer?
  - If we try to compute a value larger than the computer can store, we get an arithmetic overflow error.

CMPUT101 Introduction to Computing (c) Yngvi Bjornsson 11

Representing Unsigned Integers

- How does a 16-bit computer represent the value 14?

0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0

- What is the largest 16-bit integer?

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

$= 1x2^{15} + 1x2^{14} + \ldots + 1x2^1 + 1x2^0 = 65,535$

CMPUT101 Introduction to Computing (c) Yngvi Bjornsson 12

Representing Signed Integers

- How does a 16 bit computer represent the value -14?

1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0

Sign bit. 0: + (positive), 1: - (negative)

- What is the largest 16-bit signed integer?

0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

$= 1x2^{14} + 1x2^{13} + \ldots + 1x2^{1} + 1x2^{0} = 32{,}767$

- Problem ➔ the value 0 is represented twice!
  - Most computers use a different representation, called <u>two's complement</u>.

Representing Floating Point Numbers

- How do we represent floating point numbers like 5.75 and -143.50?
- Three step process:
  1. Convert the decimal number to a binary number.
  2. Write binary number in "normalized" scientific notation.
  3. Store the normalized binary number.
- Look at an example:
  - How do we store the number 5.75?

1. Convert decimal to binary (5.75 = ?)

…	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	…
	8	4	2	1	½	¼	

4 + 1 + ½ + ¼ = 5.75

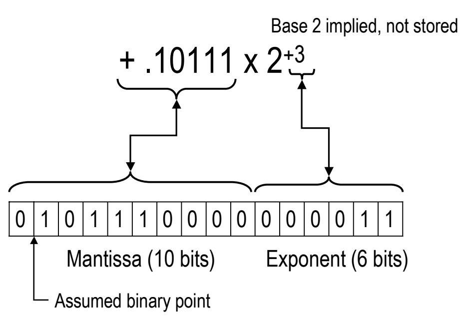
	0	1	0	1	1	1	

- 5.75 decimal ➔ 101.11 binary

2. Write using normalized scientific notation

- Scientific notation : $\pm$ M x B $^{\pm E}$
  - B is the base, M is the mantissa , E is the exponent.
  - Example: (decimal, base=10)
    - $3 = 3 \times 10^0$ (e.g. 3 * 1)
    - $2050 = 2.05 \times 10^3$ (e.g. 2.05 * 1000)
- Easy to convert to scientific notation:
  - 101.11×2^0
- Normalize to get the "." in front of first (leftmost) 1 digit
  - Increase exponent by one for each location "." moves left (decreases if we have to move left)
  - $101.11 \times 2^0 = .10111 \times 2^3$

3. Store the normalized number



Representing Text

- How can we represent text in a binary form?
  - Assign to each character a positive integer value (for example, A is 65, B is 66, …)
  - Then we can store the numbers in their binary form!
- The mapping of text to numbers ➔ Code mapping
- Need standard code mappings (why?):
  - ASCII (American Standard Code for Information Interchange) => each letter 8-bits
    - only 256 different characters can be represented (2^8)
  - Unicode => each letter 16-bits

ASCII Code mapping Table

Char	Integer	Binary	Char	Integer	Binary
	32	00100000	A	65	01000001
!	33	00100001	B	66	01000010
"	34	00100010	C	67	01000011
…	…	…	…	…	…
0	48	00110000	x	120	01111000
1	49	00110001	y	121	01111001
2	50	00110010	z	122	01111010
…	…	…	…	…	…

Example of Representing Text

- Representing the word "Hello" in ASCII
  - Look the value for each character up in the table
  - (Convert decimal value to binary)

H	e	l	l	o
72	101	108	108	111
01001000	01100101	01101100	01101100	01101111

Representing Other Information

- We need to represent other information in a computer as well
  - Pictures (BMP, JPEG, GIF, …)
  - Sound (MP3, WAVE, MIDI, AU, …)
  - Video (MPG, AVI, MP4, …)
- Different formats, but all represent the data in binary form!

Why do Computers Use Binary Numbers?

- Why not use the decimal systems, like humans?
- The main reason for using binary numbers is:
  - ➔Reliability
- Why is that?
  - Electrical devices work best in a bistable environment, that is, there are only two separate states (e.g. on/off).
  - When using binary numbers, the computers only need to represent two digits: 0 and 1

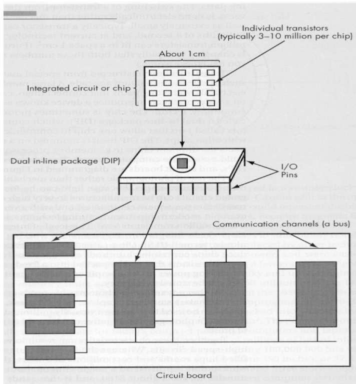
Binary Storage Devices

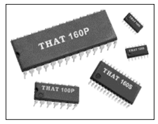
- We could, in theory at least, build a computer from any device:
  1. That has two stable states (one would represent the digit 0, the other the digit 1)
  2. Where the two states are "different" enough, such that one doesn't accidentally become the other.
  3. It is possible to sense in which state the device is in.
  4. That can switch between the two states.
- We call such devices binary storage devices
  - Can you think of any?

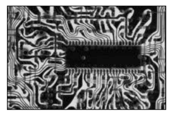
Transistor

- The binary storage device computers use is called a transistor:
  - Can be in a stable On/Off state (current flowing through or not)
  - Can sense in which state it is in (measure electrical flow)
  - Can switch between states (takes < 10 billionths of a s second!)
  - Are extremely small (can fit > 10 million/cm^2 , shrinking as we speak)
- Transistors are build from materials called semi-conductors
  - e.g. silicon
- The transistor is the elementary building block of computers, much in the same way as cells are the elementary building blocks of the human body!

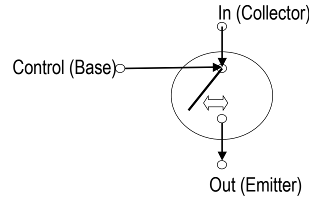
Circuit Boards, DIPs, Chips, and Transistors







Transistor – Conceptual Model



- The control line (base) is used to open/close switch:
  - If voltage applied then switch closes, otherwise is open
- Switch decides state of transistor:
  - Open: no current flowing through (0 state)
  - Closed: current flowing through (1 state)

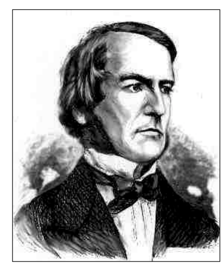
Future Development?

Why is this important?

- Transistors
  - Technology improving, allowing us to pack the transistors more and more densely (VLSI, ULSI, …)
- Can we invent more efficient binary storage devices?
  - Past: Magnetic Cores, Vacuum Tubes
  - Present: Transistors
  - Future: ?
- Quantum Computing?

Boolean Logic and Gates

Section 4.3



Boolean Logic

- Boolean logic is a branch of mathematics that deals with rules for manipulating the two logical truth values <u>true</u> and <u>false</u>.
- Named after George Boole (1815-1864)
  - An English mathematician, who was first to develop and describe a formal system to work with truth values.
- Why is Boolean logic so relevant to computers?
  - Direct mapping to binary digits!
  - 1 = true, 0 = false

Boolean Expressions

- A Boolean expression is any expression that evaluates to either true or false.
- Is the expression 1+3 a Boolean expressions?
  - No, doesn't evaluate to either true or false.
- Examples of Boolean expressions:
  - X > 100
  - X < Y
  - A = 100
  - 2 > 3

True or False ???

"This sentence is false"

Boolean Operators

- We use the three following operators to construct more complex Boolean expressions
  - AND
  - OR
  - NOT
- Examples:
  - X > 100 AND X<250
  - A=0 OR B>100

Truth Table for AND

- Let a and b be any Boolean expressions, then

a	b	a AND b
False	False	False
False	True	False
True	False	False
True	True	True

Examples X is 10 and Y is 15

X>0 AND X<20 True

X=10 AND X>Y False

Truth Table for OR

- Let a and b be any Boolean expressions, then

a	b	a OR b
False	False	False
False	True	True
True	False	True
True	True	True

Examples X is 10 and Y is 15

X>0 OR X<20 True

X=10 OR X>Y True

Truth Table for NOT

- Let a be any Boolean expression, then

a	NOT a
False	True
True	False

Examples X is 10 and Y is 15

NOT X>0 False

NOT X>Y True

Boolean Operators (cont.)

- Assume X is 10 and Y is 15.
- What is the value of the Boolean expression?
  - X=10 OR X=5 AND Y<0

(X=10 OR X=5) AND Y<0	False
X=10 OR (X=5 AND Y<0)	True

We should use parenthesis to prevent confusion!

Examples of Boolean Expressions

- Assuming X=10, Y=15, and Z=20.
- What do the following Boolean expressions evaluate to?
  - ((X=10) OR (Y=10)) AND (Z>X)
  - (X=Y) OR (NOT (X>Z))
  - NOT ((X>Y) AND (Z>Y) AND (X<Z))
  - ((X=Y) AND (X=10)) OR (Y<Z)

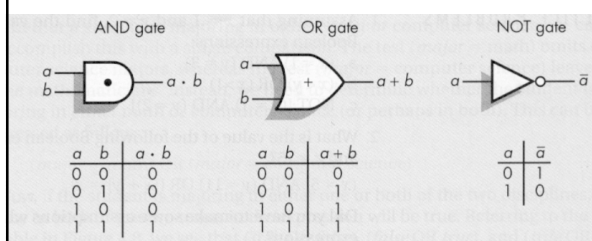
Gates

- A gate is an electronic device that operates on a collection of binary inputs to produce a binary output.
- We will look at three different kind of gates, that implement the Boolean operators:
  - AND
  - OR
  - NOT

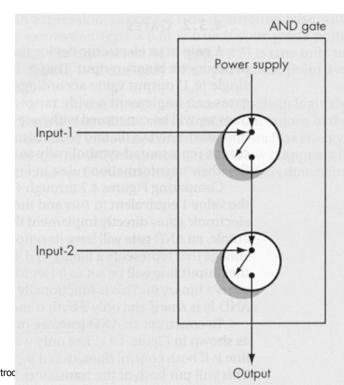
Alternative Notation

- When we are referring to gates, we use a different notation than when using Boolean expressions:
  - a AND b   a • b
  - a OR B   a + b
  - NOT a   ā
- The functionality of the operators is the same, just a different notation.

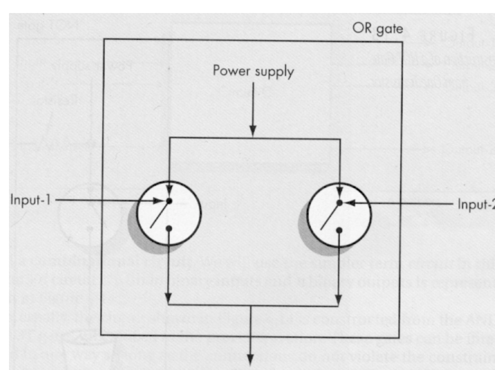
Gates AND, OR, NOT



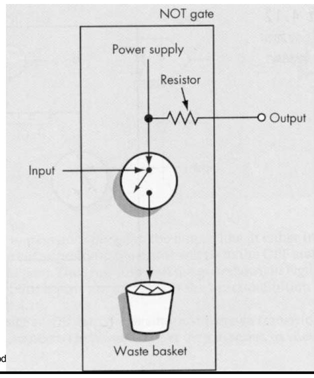
Constructing an AND Gate



Constructing an OR Gate



Constructing a NOT Gate



CMPUT101 Introd 43

Gates vs. Transistors

- We can build the AND, OR, and NOT gates from transistors.
- Now we can think of gates, instead of transistors, as the basic building blocks:
  - Higher level of abstraction, don't have to worry about as many details.
  - Can use Boolean logic to help us build more complex circuits.

CMPUT101 Introduction to Computing (c) Yngvi Bjornsson 44

Summary

- Representing information
  - External vs. Internal representation
- Computers represent information internally as
  - Binary numbers
- We saw how to represent as binary data:
  - Numbers (integers, negative numbers, floating point)
  - Text (code mappings as ASCII and Unicode)
  - (Graphics, sound, …)

CMPUT101 Introduction to Computing (c) Yngvi Bjornsson 45

Summary (cont.)

- Why do computers use binary data?
  ➔Reliability
- Electronic devices work best in a bistable environment, that is, where there are only 2 states.
- Can build a computer using a binary storage device:
  - Has two different stable states, able to sense in which state device is in, and easily switch between states.
- Fundamental binary storage device in computers:
  - Transistor

CMPUT101 Introduction to Computing (c) Yngvi Bjornsson 46

Summary (cont.)

- Boolean Logic
  - Boolean expressions are expressions that evaluate to either true or false.
  - Can use the operators AND, OR, and NOT
- Learned about gates
  - Electronic devices that work with binary input/output.
  - How to build them using transistors.
- Next we will talk about:
  - How to build circuits using gates!

CMPUT101 Introduction to Computing (c) Yngvi Bjornsson 47