COM1013
INTRODUCTION TO COMPUTER SCIENCE

Lecturer: Begüm MUTLU BİLGE, PhD

begummutlubilge+com1013@gmail.com (recommended)
bmbilge@ankara.edu.tr

CHAPTER 1

Data Storage

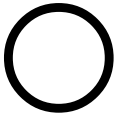
In this chapter, we consider topics associated with data representation and the storage of data within a computer. The types of data we will consider include text, numeric values, images, audio, and video. Much of the information in this chapter is also relevant to fields other than traditional computing, such as digital photography, audio/video recording and reproduction, and long-distance communication.

Bits and Their Storage

Inside today's computers, information is encoded as patterns of **0**s and **1**s.

These digits are called **bits** (short for **binary digits**).

On/true/1 **Off/false/0**

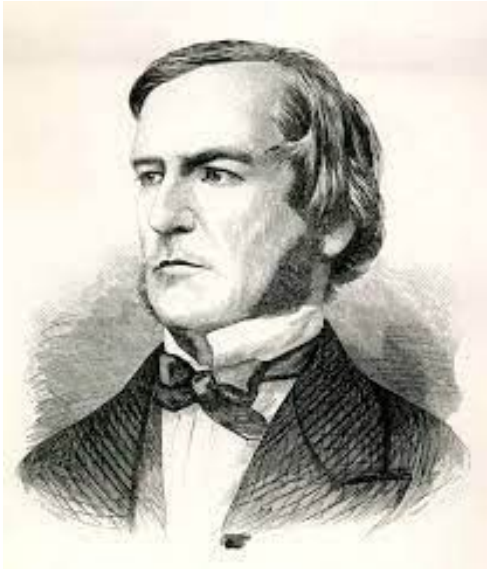


Boolean Operations

Bit **0** represents the value **false** and the bit **1** represents the value **true**

Operations that manipulate true/false values are called Boolean operations

- In honor of the mathematician George Boole (1815–1864),



Boolean Operations

In contrast to arithmetic operations, however, Boolean operations combine true/false values rather than numeric values.

The AND operation

0	0	1	1
AND 0	AND 1	AND 0	AND 1
0	0	0	1

The OR operation

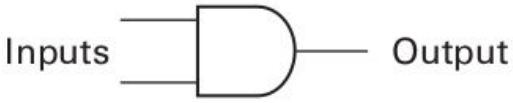
0	0	1	1
OR 0	OR 1	OR 0	OR 1
0	1	1	1

The XOR operation

0	0	1	1
XOR 0	XOR 1	XOR 0	XOR 1
0	1	1	0

Gates

AND

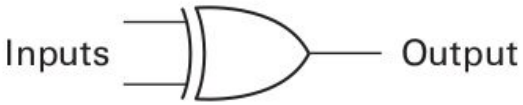


Inputs	Output
0 0	0
0 1	0
1 0	0
1 1	1

OR

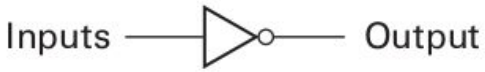
Inputs	Output
0 0	0
0 1	1
1 0	1
1 1	1

XOR



Inputs	Output
0 0	0
0 1	1
1 0	1
1 1	0

NOT

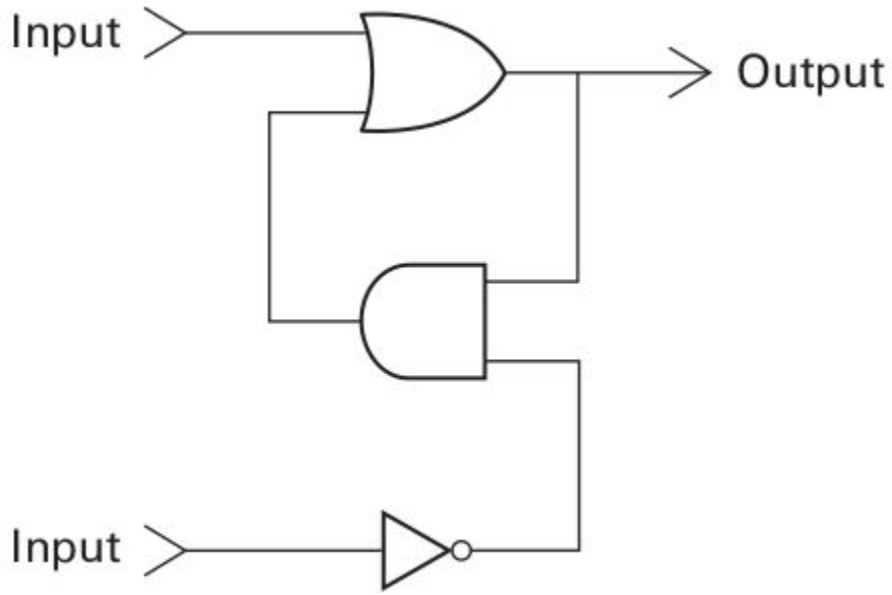


Inputs	Output
0	1
1	0

Flip-Flop

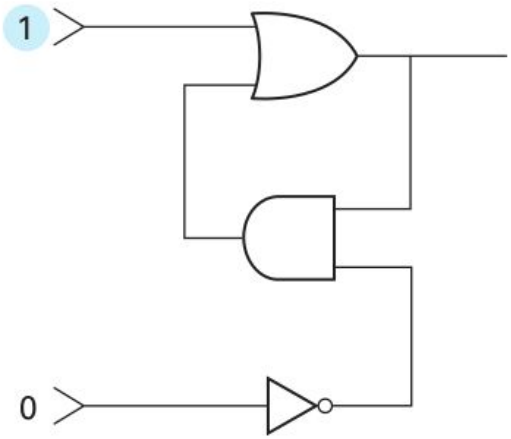
A flip-flop is a fundamental unit of computer memory.

It is a circuit that produces an output value of 0 or 1, which **remains constant until a pulse** (a temporary change to a 1 that returns to 0) **from another circuit** causes it to shift to the other value.

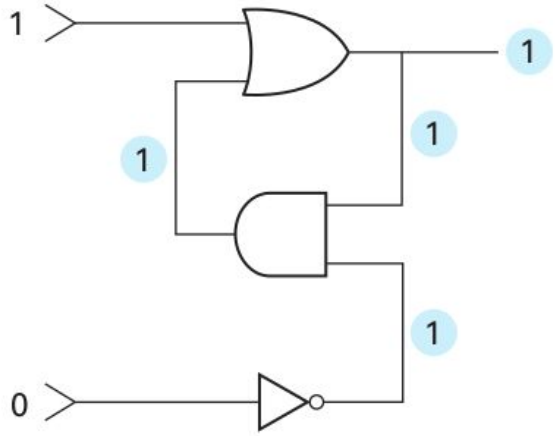


Flip-Flop

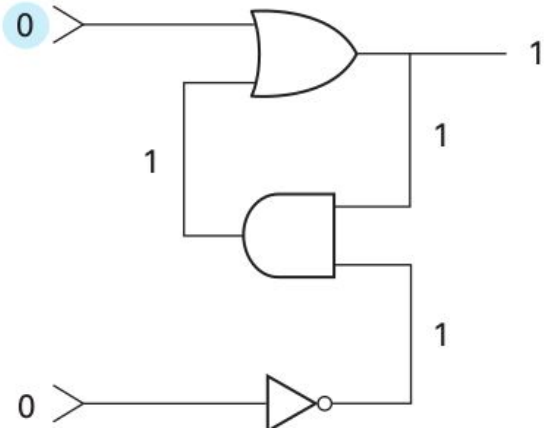
a. First, a 1 is placed on the upper input.



b. This causes the output of the OR gate to be 1 and, in turn, the output of the AND gate to be 1.



c. Finally, the 1 from the AND gate keeps the OR gate from changing after the upper input returns to 0.



Hexadecimal Notation

Some string of bits can be quite long.

- A long string of bits is often called a **stream**.
- Unfortunately, streams are difficult for the human mind to comprehend.
  - Also tedious and error prone

101101010011

10111011010100110101001110110101010011

Hexadecimal Notation

Some string of bits can be quite long.

- To simplify the representation of such bit patterns, therefore, we usually use a shorthand notation called **hexadecimal** notation,
- In particular, hexadecimal notation uses **a single symbol to represent a pattern of four bits.**
  - E.g., a string of twelve bits can be represented by three hexadecimal symbols.

Hexadecimal Encoding System

Bit pattern	Hexadecimal representation
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

Examples:

10110101 = B5

B 5

1010010011001000 = A4C8

A 4 C 8

Hexadecimal Notation

What bit patterns are represented by the following hexadecimal patterns?

a. 5FD97

b. 610A

c. ABCD

d. 0100

Use hexadecimal notation to represent the following bit patterns:

a. 0110101011110010

b. 11101000010101010100010111

c. 01001000

Main Memory

For the purpose of storing data, a computer contains a large collection of circuits (such as flip-flops),

- each capable of **storing a single bit**.

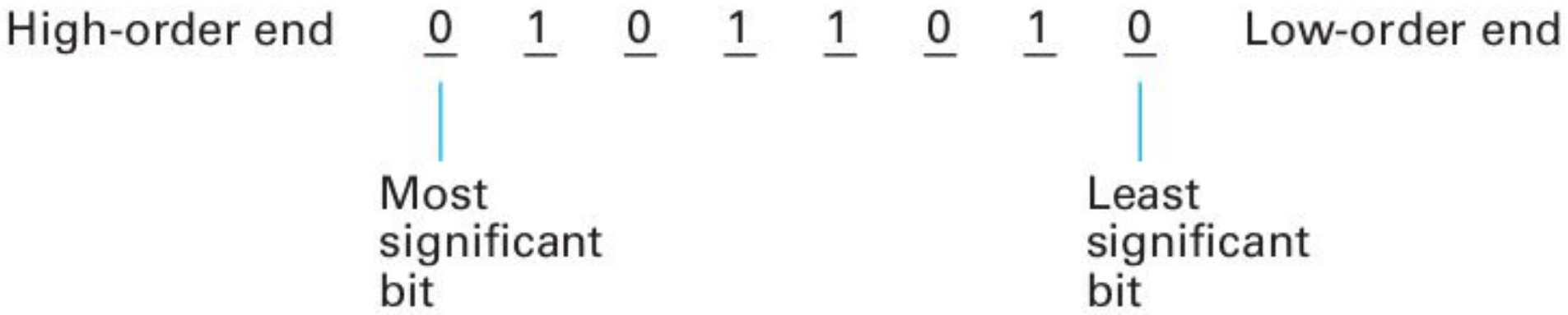
This bit reservoir is known as **the machine's main memory**.

Memory Organization

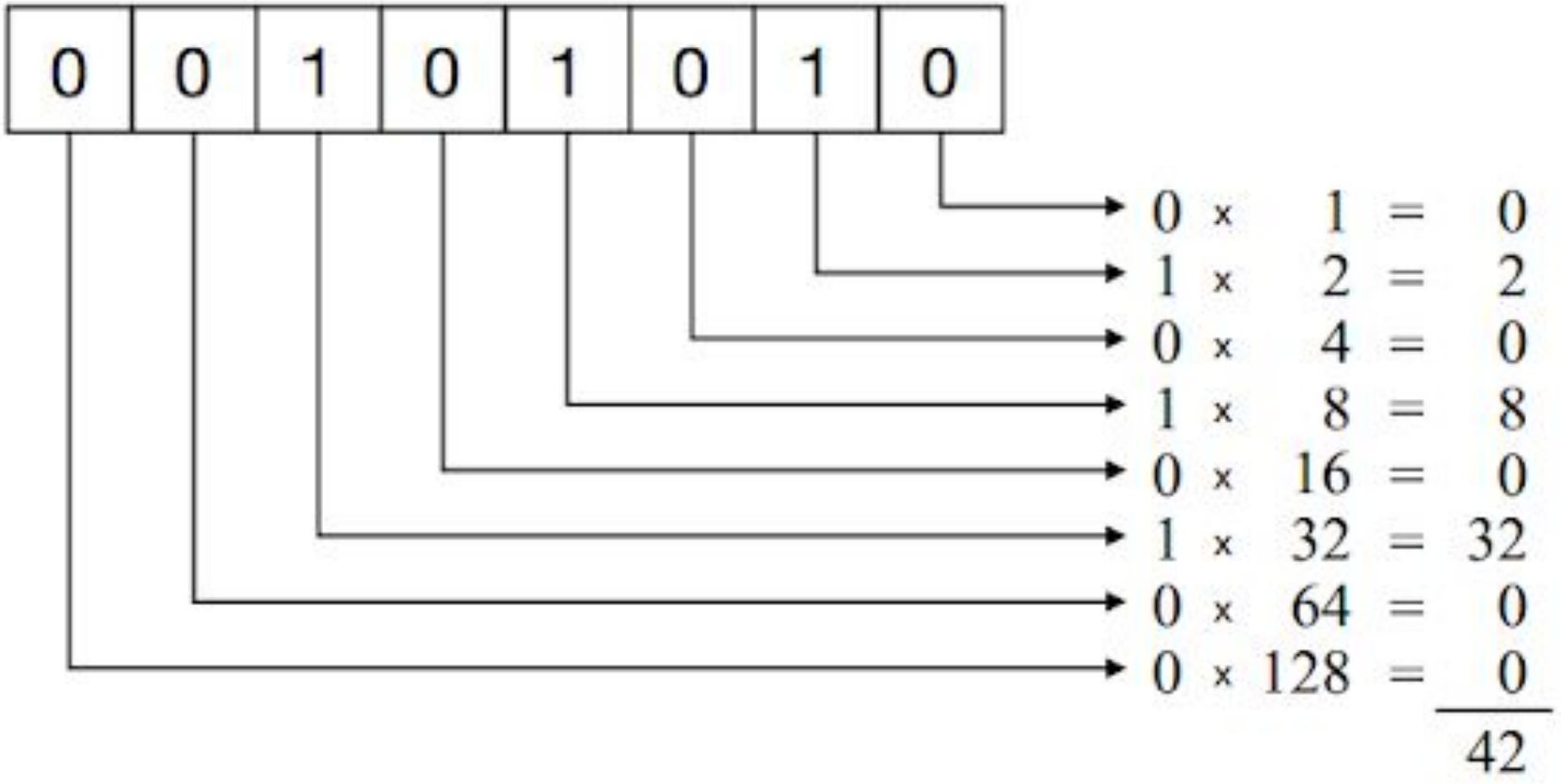
A computer's main memory is organized in manageable units called **cell**s, with a typical cell size being **eight bits**.

(A string of eight bits is called a **byte**. Thus, a typical **memory cell** has a capacity of one **byte**.)

Envision the bits within a memory cell

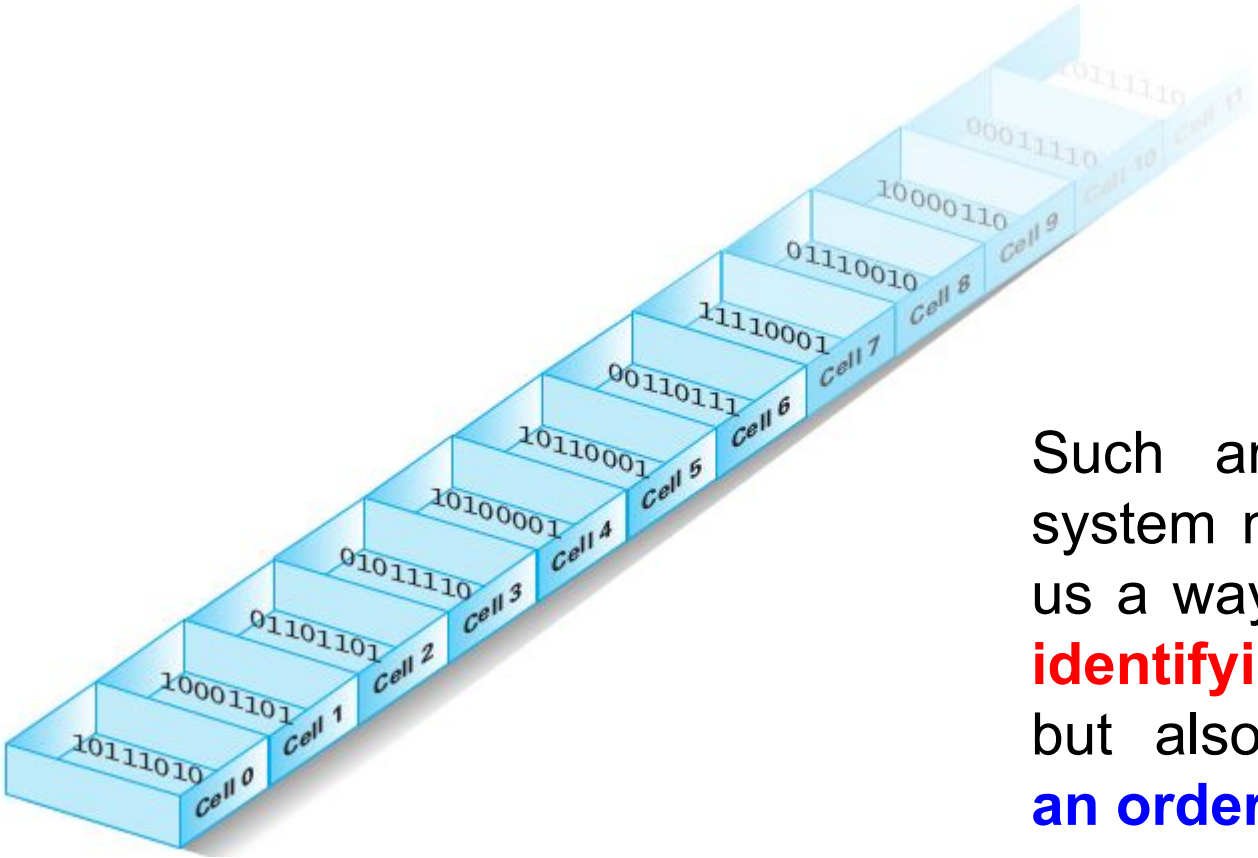


Envision the bits within a memory cell



Memory Organization

To identify individual cells in a computer's main memory, each cell is assigned a unique "name," called its **address**.



Such an addressing system not only gives us a way of **uniquely identifying each cell** but also **associates an order to the cells**.

Memory Organization

Because a computer's main memory is organized as individual, addressable cells, the cells can be **accessed independently as required**.

To reflect the ability to access cells in any order, a computer's main memory is often called **random access memory** (**RAM**).

Measuring Memory Capacity

Term	Actual	Approximate
Byte (B)	8 bits	
Kilobyte (KB)	1024 (2^{10}) Bytes	1000(10^3) Bytes
Megabyte (MB)	1024 (2^{10}) KB = 2^{20} Bytes	1000 (10^3) KB = 10^6 Bytes
Gigabyte (GB)	1024 (2^{10}) MB = 2^{30} Bytes	1000 (10^3) MB = 10^9 Bytes
Terabyte (TB)	1024 (2^{10}) GB = 2^{40} Bytes	1000 (10^3) GB = 10^{12} Bytes
Petabyte (PB)	1024 (2^{10}) TB = 2^{50} Bytes	1000 (10^3) TB = 10^{15} Bytes
Exabyte (EB)	1024 (2^{10}) PB = 2^{60} Bytes	1000 (10^3) PB = 10^{18} Bytes
Zettabyte (ZB)	1024 (2^{10}) EB = 2^{70} Bytes	1000 (10^3) EB = 10^{21} Bytes
Yottabyte (YB)	1024 (2^{10}) ZB = 2^{80} Bytes	1000 (10^3) ZB = 10^{24} Bytes

Mass Storage

Most computers have additional memory devices called mass storage (or secondary storage)

magnetic disks,

- CDs,
- DVDs,
- magnetic tapes,
- flash drives, and
- solid-state disks

Mass Storage

The advantages of mass storage systems over main memory include

- less volatility,
- large storage capacities,
- low cost, and
- in many cases, the ability to remove the storage medium from the machine for archival purposes.

Mass Storage

The major disadvantages of magnetic and optical mass storage systems

- They typically require **mechanical motion** and therefore require significantly **more time** to store and retrieve data than a machine's main memory, where all activities are performed electronically.
- Storage systems with moving parts are more prone to mechanical failures than solid state systems.

Extra Reading

Brookshear JG, Smith D, Brylow D.

“Computer science: an overview”.

Read the following section in the textbook.

Chapter: 1.3. Mass Storage

- Magnetic Systems
- Optical Systems

Summarize the section in 1 page (no figure should be included.)

Template for summarization assignments:

https://docs.google.com/document/d/1i1tU3AdIvPgRxqzGmXrsnVVCRXws5V3caDTCPMbPJ_0/edit?usp=sharing

Representing Information as Bit Patterns

How information can be encoded as bit patterns ?

Scope: encoding text, numerical data, images, and sound.

Representing Text

Information in the form of **text** is normally represented by means of a **code** in which each of the different **symbols** in the text is assigned a **unique bit pattern.**

The text is then represented as **a long string of bits** in which the **successive patterns represent the successive symbols** in the original text.

Representing Text

The American National Standards Institute (ANSI, pronounced “AN–see”)

American Standard Code for Information Interchange (ASCII, pronounced “AS–kee”).

ASCII code uses bit patterns of length 7 to represent

- the uppercase and lowercase letters of the English alphabet,
- punctuation symbols,
- the digits 0 through 9, and
- certain control information such as line feeds, carriage returns, and tabs.

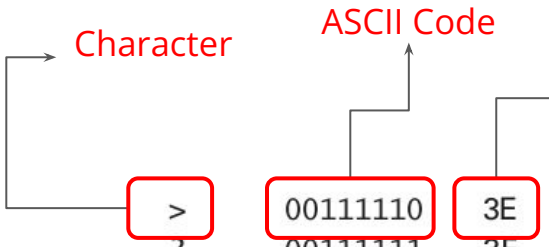
Representing Text

ASCII was extended to an 8-bit-per-symbol format by adding a 0 at the most significant end of each of the 7-bit patterns.

- To fit conveniently into a typical byte-size memory cell
- To provide 128 additional bit patterns
  - To represent symbols beyond the English alphabet and associated punctuation

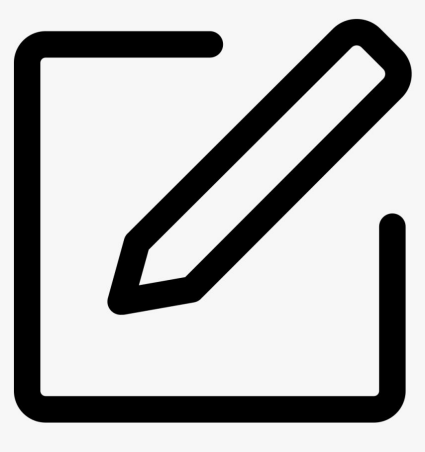
01001000	01100101	01101100	01101100	01101111	00101110
H	e	l	l	o	.

Representing Text



line feed	00001010	0A	>	00111110	3E	^	01011110	5E
carriage return	00001011	0B	?	00111111	3F	_	01011111	5F
space	00100000	20	@	01000000	40	`	01100000	60
!	00100001	21	A	01000001	41	a	01100001	61
”	00100010	22	B	01000010	42	b	01100010	62
#	00100011	23	C	01000011	43	c	01100011	63
$	00100100	24	D	01000100	44	d	01100100	64
%	00100101	25	E	01000101	45	e	01100101	65
&	00100110	26	F	01000110	46	f	01100110	66
’	00100111	27	G	01000111	47	g	01100111	67
(	00101000	28	H	01001000	48	h	01101000	68
)	00101001	29	I	01001001	49	i	01101001	69
*	00101010	2A	J	01001010	4A	j	01101010	6A
+	00101011	2B	K	01001011	4B	k	01101011	6B
’	00101100	2C	L	01001100	4C	l	01101100	6C
.	00101101	2D	M	01001101	4D	m	01101101	6D
/	00101110	2E	N	01001110	4E	n	01101110	6E
0	00101111	2F	O	01001111	4F	o	01101111	6F
1	00110000	30	P	01010000	50	p	01110000	70
2	00110001	31	Q	01010001	51	q	01110001	71
3	00110010	32	R	01010010	52	r	01110010	72
4	00110011	33	S	01010011	53	s	01110011	73
5	00110100	34	T	01010100	54	t	01110100	74
6	00110101	35	U	01010101	55	u	01110101	75
7	00110110	36	V	01010110	56	v	01110110	76
8	00110111	37	W	01010111	57	w	01110111	77
9	00111000	38	X	01011000	58	x	01111000	78
:	00111001	39	Y	01011001	59	y	01111001	79
;	00111010	3A	Z	01011010	5A	z	01111010	7A
<	00111011	3B	[	01011011	5B	{	01111011	7B
=	00111100	3C	\	01011100	5C	|	01111100	7C
	00111101	3D	]	01011101	5D	}	01111101	7D

Representing Text



- **Text files**
  - Simple text files
  - Text editors
  - Word processors

Representing Image (as Bit Map)

The image: as a collection of dots

- Each of which is called a pixel, short for “picture element.”

Black-white images

- Pixel⇛Bit (0 or 1)

Gray-scale images

- Pixel⇛Byte (allows a variety of shades of grayness to be represented)

Color images

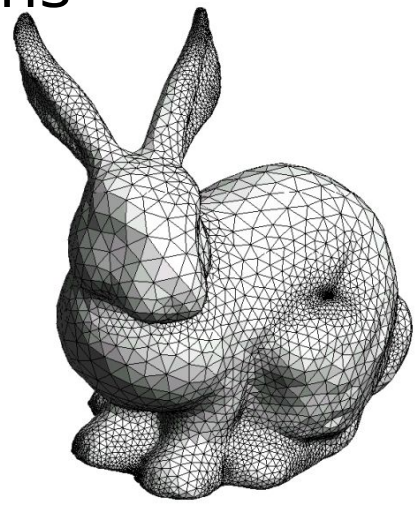
- RGB encoding (3 color components)
- “brightness” component and 2 color components

Representing Image (as Geometric Structure)

The image: as a collection of geometric structures, such as lines and curves, that can be encoded using techniques of analytic geometry

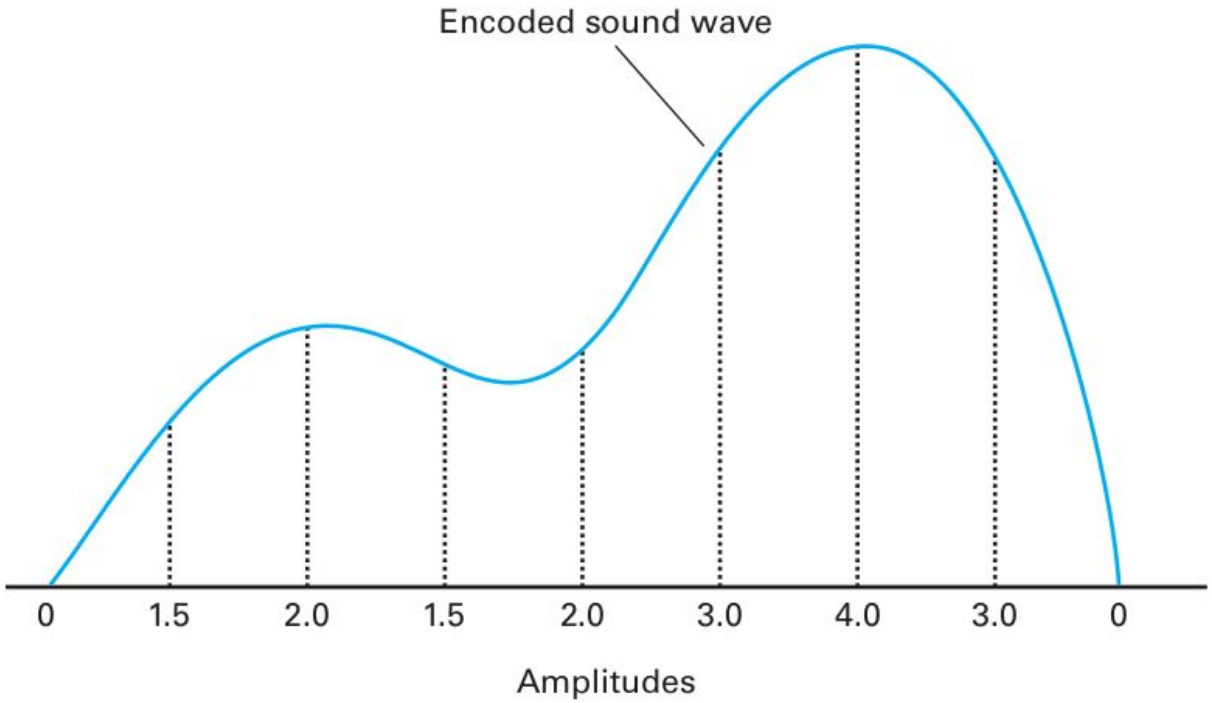
Applicable in:

- The scalable fonts that are available via today's word processing systems
- Computer-aided design (CAD) systems



Representing Sound

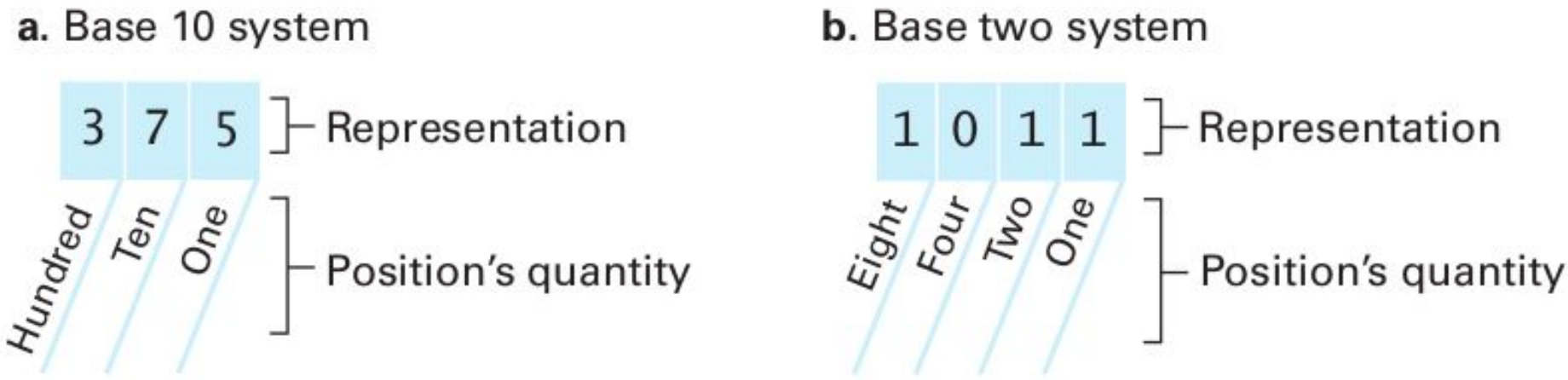
The most generic method of encoding audio information for computer storage and manipulation is to sample the **amplitude of the sound wave** at regular intervals and record the series of values obtained.



Representing Numeric Values

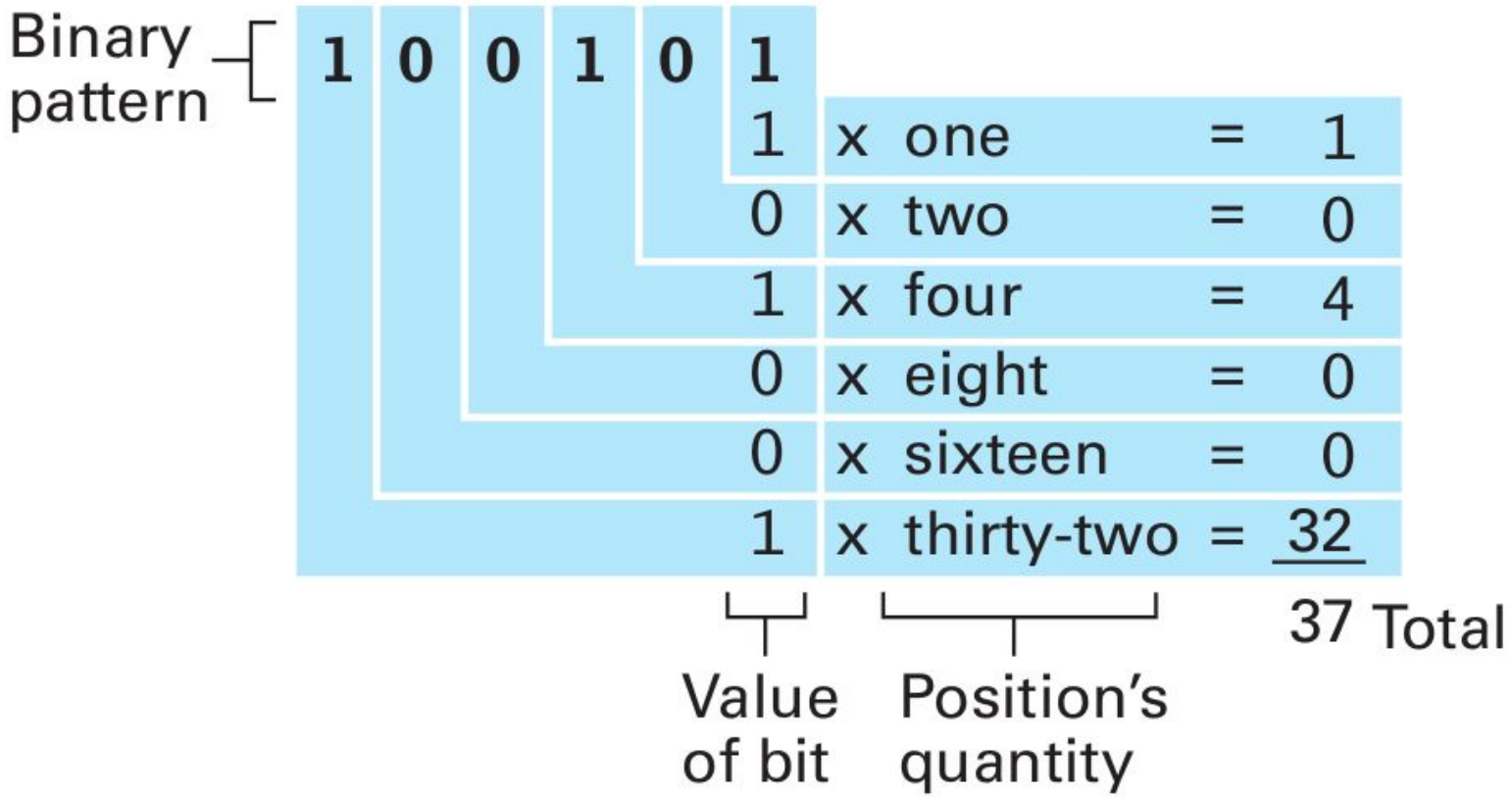
Binary notation (or variations of it) is used extensively for encoded numeric data for computer storage

The base 10 and binary systems



Binary Notation

Decoding the binary representation



Binary Notation

An algorithm for finding the binary representation of a positive integer

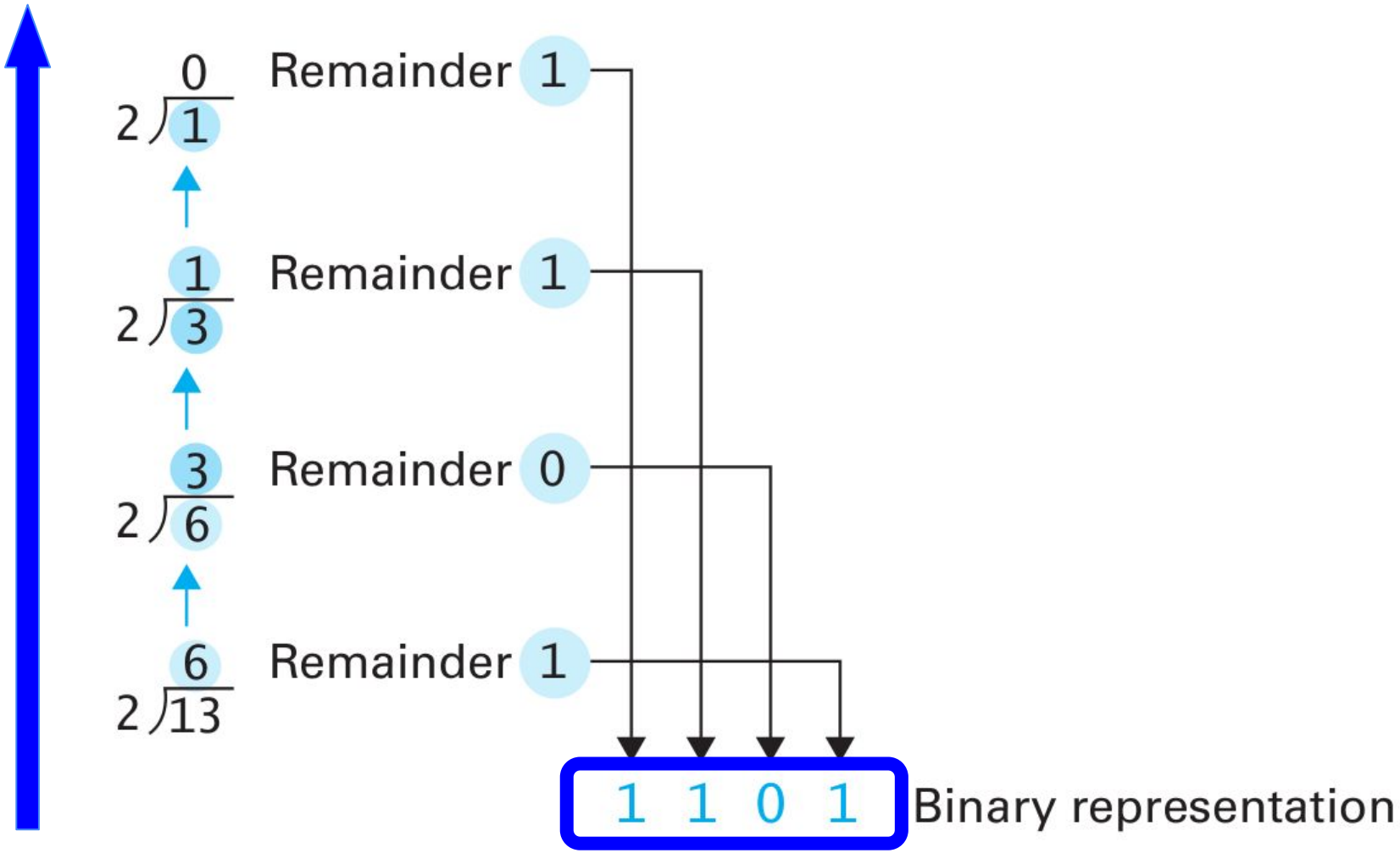
Step 1. Divide the value by two and record the remainder.

Step 2. As long as the quotient obtained is not zero, continue to divide the newest quotient by two and record the remainder.

Step 3. Now that a quotient of zero has been obtained, the binary representation of the original value consists of the remainders listed from right to left in the order they were recorded.

Binary Notation

Applying the algorithm (in prev. slide) to obtain the binary representation of 13.



Binary Addition

To understand the process of adding two integers that are represented in binary,

- Lets first recall the process of adding values that are represented in traditional base 10 notation
  - **Example**

$$\begin{array}{r} 58 \\ +\ 27 \\ \hline \end{array} \qquad \begin{array}{r} 1 \\ 58 \\ +\ 27 \\ \hline 5 \end{array} \qquad \begin{array}{r} 58 \\ +\ 27 \\ \hline 85 \end{array}$$

Binary Addition

Binary Addition Facts

$$\begin{array}{r} 0 \\ +0 \\ \hline 0 \end{array} \qquad \begin{array}{r} 1 \\ +0 \\ \hline 1 \end{array} \qquad \begin{array}{r} 0 \\ +1 \\ \hline 1 \end{array} \qquad \begin{array}{r} 1 \\ +1 \\ \hline 10 \end{array}$$

To add two integers represented in binary notation, we follow the same procedure except that all sums are computed using the addition facts (above).

Binary Addition

Example

```
                    1          1          1
  111010      111010     111010     111010      111010
+  11011    +  11011   +  11011   +  11011    +  11011
---------   --------   --------   --------   ---------
                  01       0101     010101     1010101
```

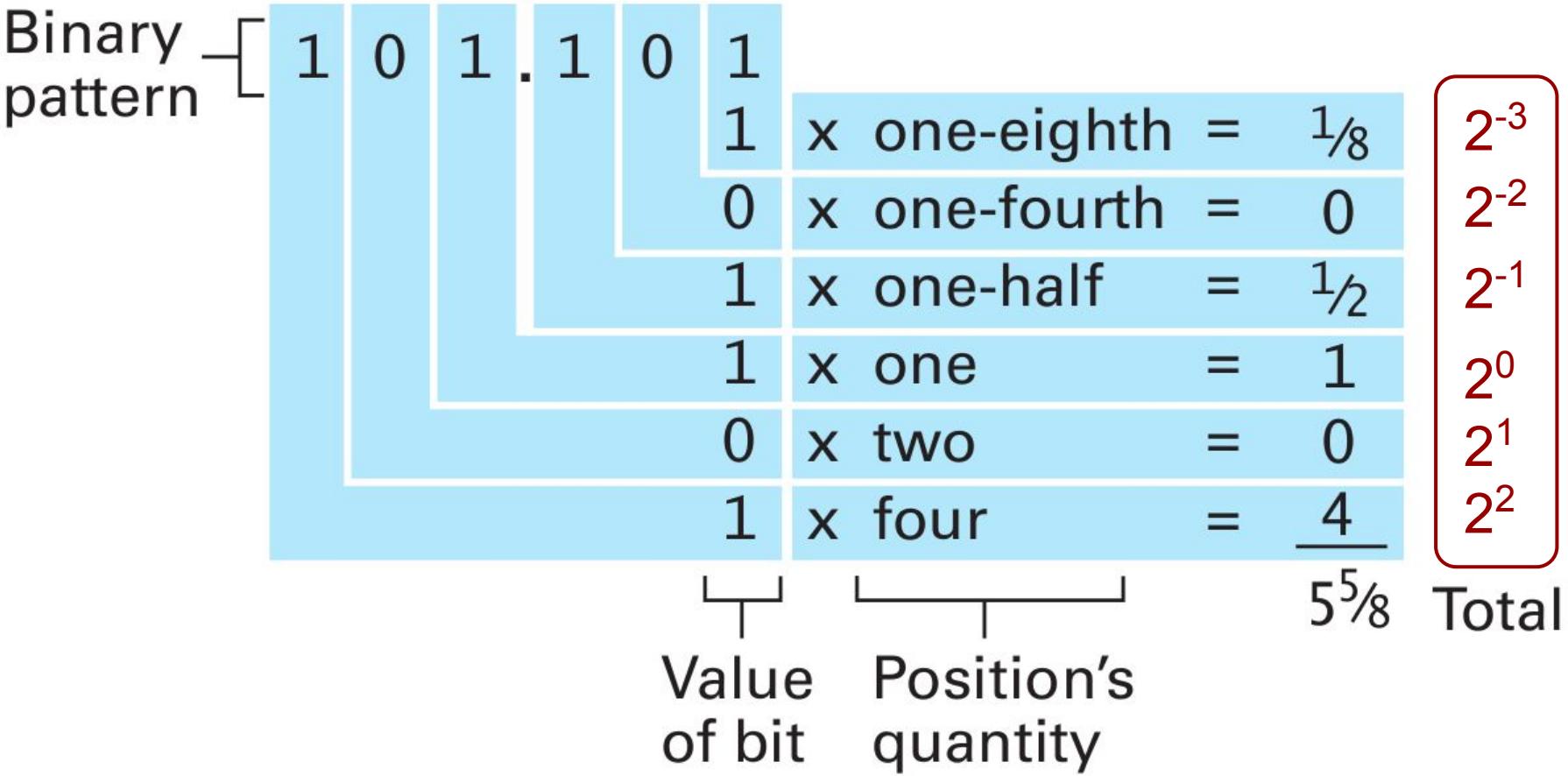
More Practice

111111 + 111 = ?

101100 + 101011 = ?

10110011 + 11101011 = ?

Fractions in Binary




```
  10.011
+ 100.110
---------
 111.001
```

To add two binary representations having radix points, we merely align the radix points and apply the regular addition process.

Fractions in Binary

$$\begin{array}{r} 10.011 \\ +\ 100.110 \\ \hline 111.001 \end{array}$$

To add two binary representations having radix points, we merely align the radix points and apply the regular addition process.

Storing Integers

The most popular 2 systems for representing integers:

- Two’s complement notation
- Excess notation

Two's complement notation

The most popular system for **representing integers within today's computers**.

A **fixed number of bits** to represent each of the values in the system.

In today's equipment, it is common to use a two's complement system in which

- each value is represented by a pattern of **32 bits**.

Two's complement notation

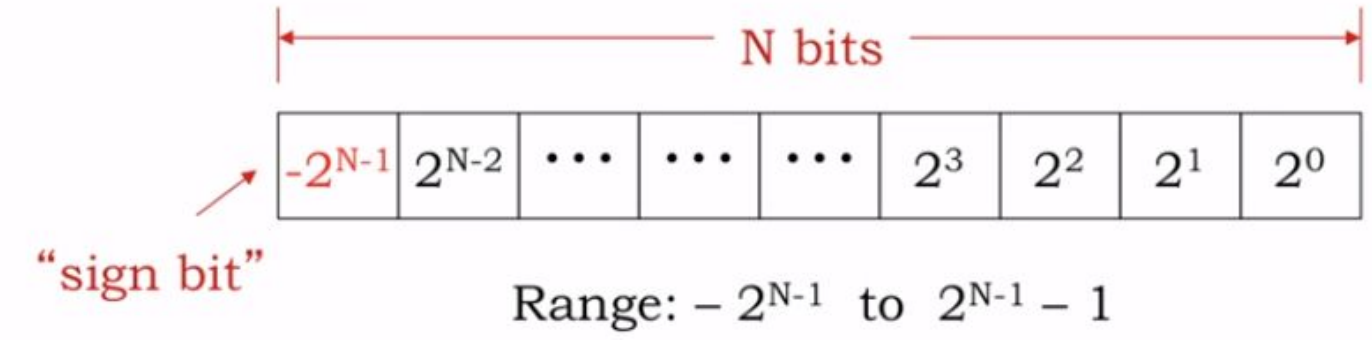
The leftmost bit of a bit pattern indicates the sign of the value represented.

- Thus, the leftmost bit is often called the **sign bit**.

In a two's complement system, **negative** values are represented by the patterns whose **sign bits are 1**; **nonnegative** values are represented by patterns whose **sign bits are 0**.

Two's complement notation

In a two's complement encoding, the high-order bit of the N-bit representation has negative weight.



Negative numbers have "1" in the high-order bit.

Most negative number : 10...0000 -2^{N-1}

Most positive number : 10...0000 $+2^{N-1}-1$

If all bits are "1" : 11...1111 -1

If all bits are "0" : 00...0000 0

Two's complement notation

a. Using patterns of length three

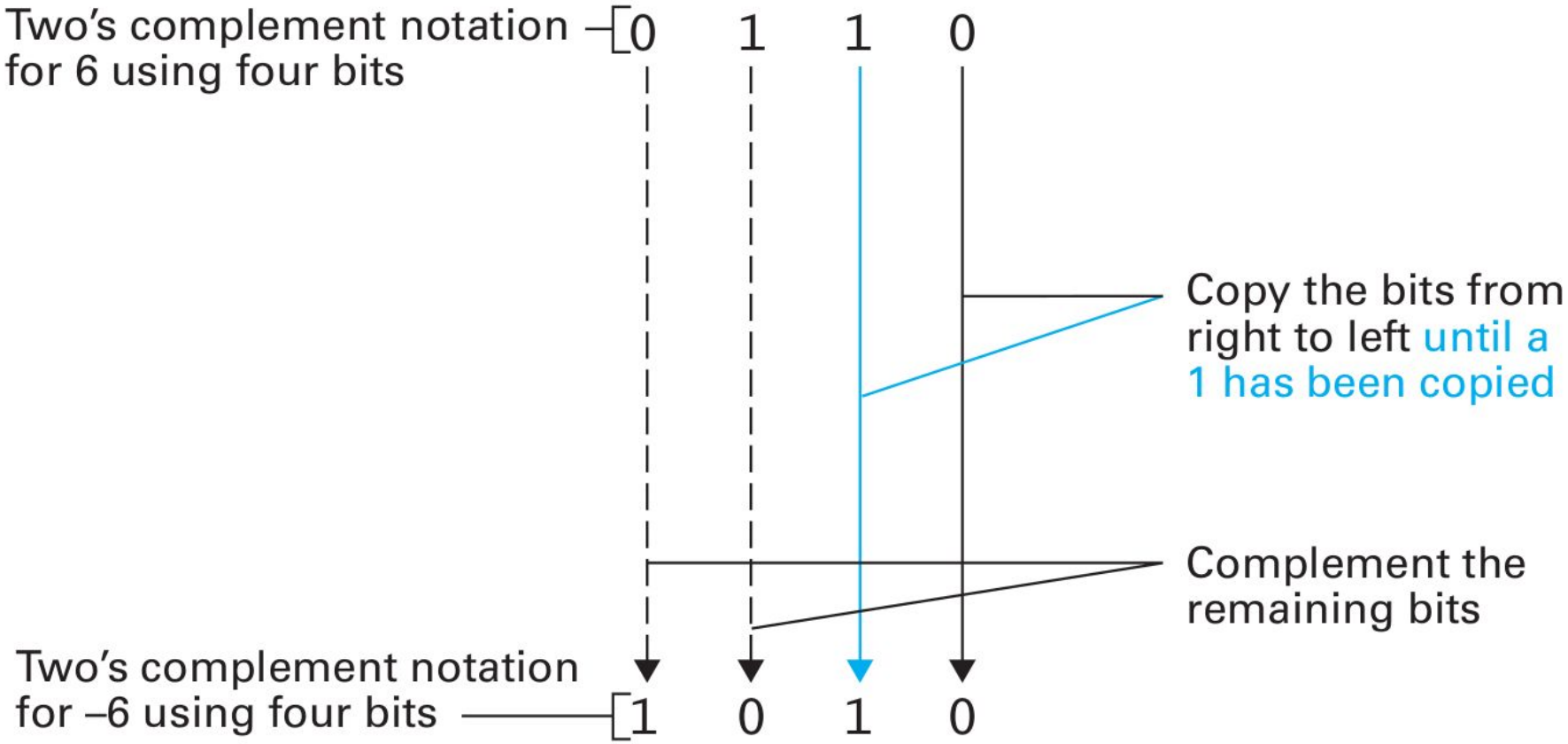
Bit pattern	Value represented
011	3
010	2
001	1
000	0
111	-1
110	-2
101	-3
100	-4

b. Using patterns of length four

Bit pattern	Value represented
0111	7
0110	6
0101	5
0100	4
0011	3
0010	2
0001	1
0000	0
1111	-1
1110	-2
1101	-3
1100	-4
1011	-5
1010	-6
1001	-7
1000	-8

Two's complement notation

Encoding the value -6 in two's complement notation using 4 bits



Two's complement notation

Addition in Two's Complement Notation

To add values, we apply the same algorithm that we used for binary addition,

- except that **all bit patterns**, including the answer, **are the same length**.

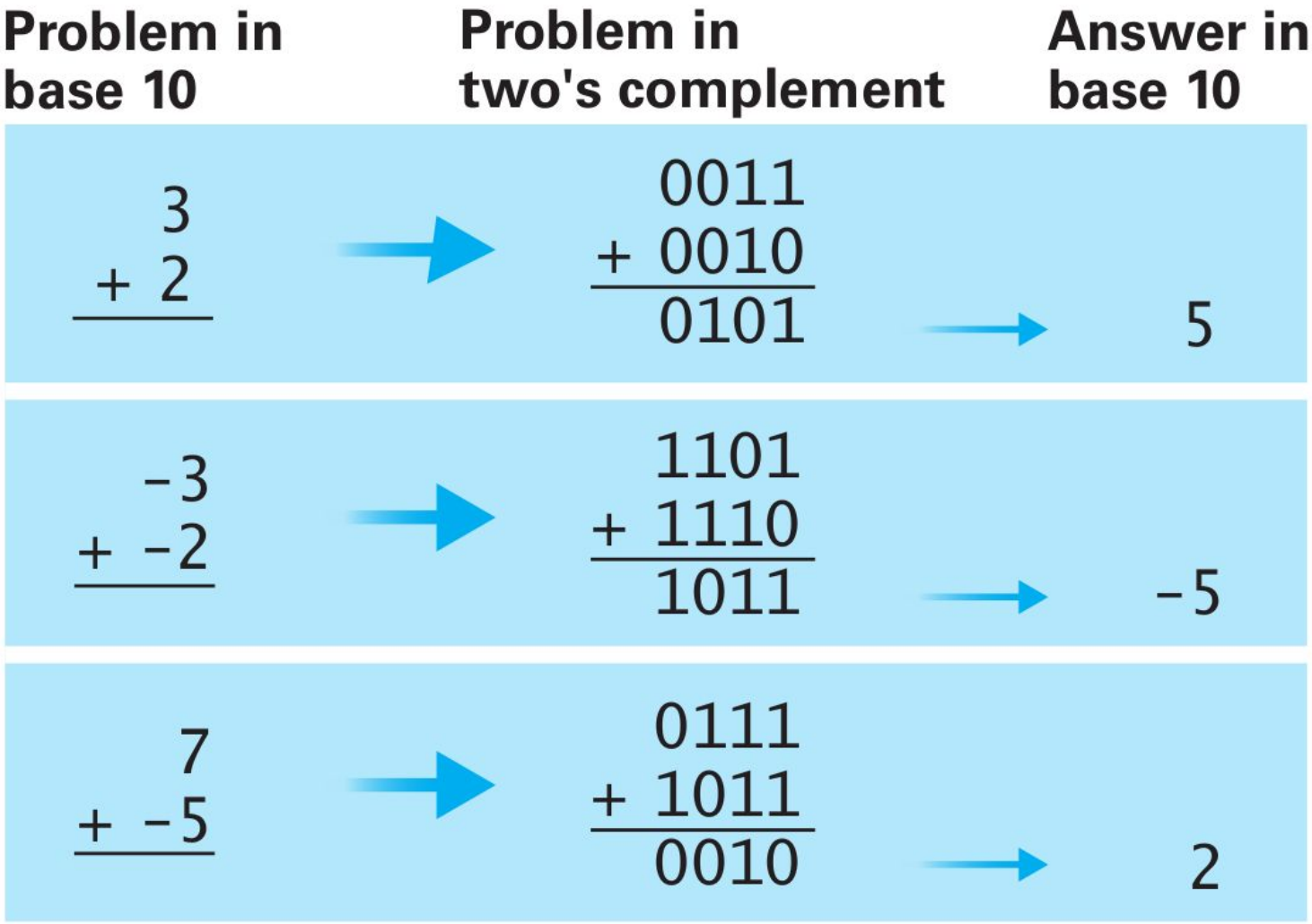
This means that when adding in a two's complement system, any extra bit generated on the left of the answer by **a final carry must be truncated**.

Thus "adding" 0101 and 0010 produces 0111, and "adding" 0111 and 1011 results in 0010

- (0111 + 1011 = 10010, which is truncated to 0010).

Two's complement notation

Addition in Two's Complement Notation

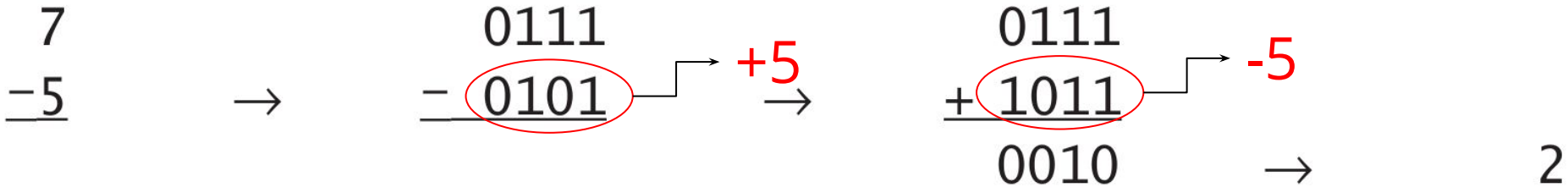


Two's complement notation

For example, the subtraction problem 7 - 5 is the same as the addition problem 7 + (-5).

Consequently, if a machine were asked to subtract 5 (stored as 0101) from 7 (stored as 0111),

- it would first change the 5 to -5 (represented as 1011) and then
- perform the addition process of 0111 + 1011 to obtain 0010, which represents 2, as follows:



The Problem of Overflow

Overflow is the problem that occurs when a computation produces a value that falls outside the range of values that can be represented.

When using two's complement with patterns of 4 bits, the largest positive integer that can be represented is 7, and the most negative integer is -8.

- 4+5 = ?
  - the result would appear as -7. 😕

```
  0 1 0 0    (+4)
+ 0 1 0 1    (+5)
-------------
  1 0 0 1    (-7)
```

The Problem of Overflow

- How to detect overflow ?
  - by checking the sign bit of the answer.
    - An overflow is indicated
      - If the addition of two positive values results in the pattern for a negative value or
      - If the sum of two negative values appears to be positive.

The Problem of Overflow

Most computers use two's complement systems with longer bit patterns than we have used in our examples

- Larger values can be manipulated without causing an overflow.

Today, it is common to use patterns of 32 bits for storing values in two's complement notation,

- allowing for positive values as large as 2,147,483,647 to accumulate before overflow occurs.

If still larger values are needed,

- longer bit patterns can be used or
- perhaps the units of measure can be changed.

Excess Notation

Bit pattern	Value represented
1111	7
1110	6
1101	5
1100	4
1011	3
1010	2
1001	1
1000	0
0111	-1
0110	-2
0101	-3
0100	-4
0011	-5
0010	-6
0001	-7
0000	-8

As is the case with two's complement notation,

- Each of the values in an excess notation system is represented by a bit pattern of the same length.

Bit pattern	Value represented
111	3
110	2
101	1
100	0
011	-1
010	-2
001	-3
000	-4

Research Themes

Data Compression

(See “Computer science: an overview” Chapter 1.9)

Communication Errors

(See “Computer science: an overview” Chapter 1.10)

Storage Devices: HDD vs. SSD

COM1013
INTRODUCTION TO COMPUTER SCIENCE

Lecturer: Begüm MUTLU BİLGE, PhD

begummutlubilge+com1013@gmail.com (recommended)
bmbilge@ankara.edu.tr